



ENGINEERING & INDUSTRIAL RESEARCH STATION

ELECTRICAL ENGINEERING / MISSISSIPPI STATE UNIVERSITY

(NASA-CR-150392) COMPENSATOR IMPROVEMENT
FOR MULTIVARIABLE CONTROL SYSTEMS Final
Report, 25 Aug. 1976 - 25 Aug. 1977
(Mississippi State Univ., Mississippi
State.) 154 p HC A08/MF A01

N77-31864

Unclass

CSCL 09B G3/63 47702

**COMPENSATOR IMPROVEMENT
FOR MULTIVARIABLE CONTROL SYSTEMS**

by JERREL R. MITCHELL,
WILLIE L. McDANIEL, Jr.
& LENNOR LYNN GRESHAM



MSSU-EIRS-EE-78-1

COLLEGE OF ENGINEERING ADMINISTRATION

HARRY C. SIMRALL, M.S.
DEAN, COLLEGE OF ENGINEERING

WILLIE L. MCDANIEL, JR., PH.D.
ASSOCIATE DEAN

WALTER R. CARNES, PH.D.
ASSOCIATE DEAN

LAWRENCE J. HILL, M.S.
DIRECTOR, ENGINEERING EXTENSION

CHARLES B. CLIETT, M.S.
AEROPHYSICS & AEROSPACE ENGINEERING

WILLIAM R. FOX, PH.D.
AGRICULTURAL & BIOLOGICAL ENGINEERING

JOHN L. WEEKS, JR., PH.D.
CHEMICAL ENGINEERING

ROBERT M. SCHOLTES, PH.D.
CIVIL ENGINEERING

B. J. BALL, PH.D.
ELECTRICAL ENGINEERING

W. H. EUBANKS, M.ED.
ENGINEERING GRAPHICS

FRANK E. COTTON, JR., PH.D.
INDUSTRIAL ENGINEERING

C. T. CARLEY, PH.D.
MECHANICAL ENGINEERING

JOHN I. PAULK, PH.D.
NUCLEAR ENGINEERING

ELDRED W. HOUGH, PH.D.
PETROLEUM ENGINEERING

For additional copies or information
address correspondence to

ENGINEERING AND INDUSTRIAL RESEARCH STATION
DRAWER DE
MISSISSIPPI STATE UNIVERSITY
MISSISSIPPI STATE, MISSISSIPPI 39762

TELEPHONE (601) 325-2266



Mississippi State University does not discriminate on the grounds of race, color, religion, sex, or national origin

Under the provisions of Title IX of the Educational Amendments of 1972, Mississippi State University does not discriminate on the basis of sex in its educational programs or activities with respect to admissions or employment. Inquiries concerning the application of these provisions may be referred to Dr. T. K. Martin, Vice President, 610 Allen Hall, Drawer J, Mississippi State, Mississippi 39762, or to the Director of the Office for Civil Rights of the Department of Health, Education and Welfare.



MISSISSIPPI STATE UNIVERSITY
DEPARTMENT OF ELECTRICAL ENGINEERING

COMPENSATOR IMPROVEMENT FOR
MULTIVARIABLE CONTROL SYSTEMS

For Period Covering
August 25, 1976 to August 25, 1977

Final Report
NAS8-31568

Principal Investigators:
Jerrel R. Mitchell, Ph.D.
Willie L. McDaniel, Jr., Ph.D.

Prepared by:
Lennor Lynn Gresham

Prepared for:
George C. Marshall Space Flight Center
Marshall Space Flight Center, AL 35812

ABSTRACT

The objective of this research is to develop the theory and associated numerical technique for the iterative design improvement of the compensation for linear, time-invariant control systems with multiple inputs and multiple outputs. The multivariable capabilities allow system suboptimization of several control loops with coupled characteristics. A strict constraint algorithm is used in obtaining a solution of the specified constraints of the control design. The result of the research effort is the Multiple Input, Multiple Output Compensator Improvement Program (CIP).

The objective of the Compensator Improvement Program is to modify in an iterative manner the free parameters of the dynamic compensation matrix so that the system satisfies frequency domain specifications. In this exposition, the underlying principles of the multivariable CIP algorithm are presented and the practical utility of the program is illustrated with space vehicle related examples. Further, the capabilities of and possible extensions to the algorithm are delineated.

TABLE OF CONTENTS

Chapter	Page
ABSTRACT	ii
LIST OF FIGURES	v
LIST OF TABLES	vi
SYMBOLS	vii
I. INTRODUCTION AND LITERATURE SURVEY	1
Introduction	1
Literature Survey of Previous Work	4
The Research Area	15
II. DESCRIPTION OF THE DESIGN ALGORITHM	16
An Overview of the Design Problem	16
The Algorithm Design Philosophy	18
Theoretical Concepts Associated with the Algorithm	20
III. DESCRIPTION OF THE COMPUTATIONAL FLOW DIAGRAM	27
Implementation of the Design Algorithm	27
Section A: Data Input	28
Section B: Frequency Response Manipulations	33
Section C: Evaluation of the Continuance Criterion	36
Section D: Determination of the Gradient Vectors	38
Section E: Calculation of the Directional Vector	42
IV. INVESTIGATIONS AND EXAMPLES	47

TABLE OF CONTENTS (Continued)

Chapter	Page
V. CONCLUSIONS AND RECOMMENDATIONS	59
APPENDICES	64
Appendix A: Subprogram Summaries of the Compensator Improvement Program . . .	65
Appendix B: Fortran IV Listing of the Compensator Improvement Program . . .	110
REFERENCES	144

LIST OF FIGURES

Figure		Page
1.	General Configuration for a Single Control Input System	6
2.	A Multivariable Control System	17
3.	Computational Flow Diagram of the CIP Algorithm . .	19
4.	Vector Representation of the Multivariable Control System	22
5.	Logic Diagram Representing the Frequency Response Manipulations of Section B	34
6.	Logic Diagram of the Continuance Criterion of Section C	37
7.	Logic Diagram Representing the Calculation of the Gradient Vectors of Section D	39
8.	Logic Diagram Representing the Calculation of the Directional Vector and Corresponding Compensator Enhancement of Section E	43
9.	s-Plane Configuration Displaying the Sector of Desirable Compensator Pole Locations and Typical Pole Locations for a Second Order System	45
10.	Block Diagram Representing Each Subsystem of the Finned Vehicle Example	48
11.	Frequency Response Representing Each Subsystem of the Uncompensated System of the Finned Vehicle Example	50
12.	The Improved Compensated Frequency Response of the Finned Vehicle Example	50

LIST OF TABLES

Table	Page
1. General Input Data Information	29
2. Frequency Response Input Data Describing the Plant in the Finned Vehicle Example	48
3. System Specifications for the Finned Vehicle Example	51
4. Frequency Response Data Describing the Coupling Elements of the Plant Matrix $[P(s)]$ in the Coupled Finned Vehicle Example	53
5. Plant Dynamics Describing the Yaw/Roll Ascent Flight Control System for the Space Shuttle Example	55
6. Compensation Matrix $[G(s)]$ in Cascaded Factor Form for the Space Shuttle Example	56
7. System Specifications for the Space Shuttle Example	58

SYMBOLS

Abbreviations

AM	Attenuation margin
CIA	Constraint Improvement Algorithm [7]
CIP	Compensator Improvement Program
GM	Gain margin
PM	Phase margin
SM	Stability margin
SIFR, TIFR	Sum improved frequency response, Total improved frequency response

Symbols

α	Point at which the general feedback system is broken in obtaining the open-loop frequency response
$d(\omega)$	Distance from some $GH(j\omega)$ frequency point to another point in the plane
$[G(s)]$	A $N \times M$ matrix transfer function describing the cascaded compensation and control law
$[P(s)]$	A $M \times N$ matrix describing the open loop frequency response of the plant
$[C(s)]$	A $N \times 1$ output vector
$C_k(s)$	Represents the k th subsystem output vector
$[E(s)]$	A $N \times 1$ error vector
$[R(s)]$	A $N \times 1$ input vector
$[X(s)]$	A $N \times 1$ vector representing the sensed or measured states
$[H(s)]$	A $N \times N$ unity matrix
I	A unity matrix
$GH(j\omega)$	Open-loop frequency response

SYMBOLS (Continued)

Symbols

$\sum$	Summation
π	Product
$\text{RE}[z]$	Real part of the complex variable z
$\text{IM}[z]$	Imaginary part of the complex variable
$\nabla V(x)$	The gradient of the function $V(x)$
ϵ	Belongs to or element of
$\frac{\partial d}{\partial w}$	Partial of d with respect to a parameter w
<u>—</u>	Indicates the underlined quantity is a vector
[]	Indicates the enclosed quantity is a matrix
	The magnitude of a scalar; the determinant of a matrix
j	The imaginary operator when it does not appear as a subscript
s	The Laplace transform variable representing the complex frequency $\sigma + j\omega$; for convenience, used interchangeably with $j\omega$ indicating the real frequency $\sigma = 0$ when applicable
*	Denotes the complex conjugate

CHAPTER I

INTRODUCTION AND SURVEY OF PREVIOUS WORK

Introduction

The space age era has challenged the control theorist to examine the fastest and most efficient means of system design and analysis. In this regard, increasing emphasis has been stressed on the utilization of digital computers in modern control theory. Much has been written on employing digital computer control methods for single input systems, but the complexity of systems may require more than this simple approach--that is, for a better control the sensing of many system parameters and inputs as well as their relationships to one another must determine the control law. The facet of control theory requires exploiting multiple input, multiple output techniques. Further, the problem of producing the 'best' compensated system becomes one of satisfying physical restrictions of system parameters as well as digital computer limitations.

Modern trends in engineering systems are toward greater complexity, due mainly to the requirements of complex tasks and the necessity for accuracy. Sophisticated technology involves systems that are described adequately only by numerous variables; thus, these high-order complex systems are responsible for the dichotomy of the academic and industrial attitudes. The naivete of low-order system textbook orientation must be abandoned, and with the essence

of the classical methods espoused, analysis of large-scale systems made possible with modern methods. The necessity of meeting increasingly stringent requirements on the performance of control systems, the increase in system complexity, and the accessibility of large-scale computers has forced modern theorists to reexamine the problem of interaction of multiple control inputs. A complex system may have many inputs and outputs which may be interrelated in a complicated manner. To analyze such a system, it is essential to reduce the complexity of the mathematical expressions as well as to resort to computer algorithms for the tedious computations.

Many process control systems have multiple inputs and outputs. Generally, there is no assurance that changes in one reference input will affect only one output; thus, the inputs and outputs are not decoupled but interact with one another. Analysis of such interactions of all inputs with all outputs is a difficult task, particularly by classical means. It is for this reason that the investigation and development of a computerized algorithm for compensation of multiple input, multiple output systems with interactions of parameters are so important.

In conventional control theory, generally only the input, output, and error signals are of concern; the design and analysis are developed using transfer functions, together with a variety of classical techniques such as root locus, Nyquist, Bode, etc. The appealing characteristic of conventional control theory is that it is based on the input-output relationships of the system; that is, the transfer function.

The main disadvantage of conventional theory is that, primarily, it is applicable only to linear time-invariant systems having a single input and output. It is difficult to apply to time-varying systems, nonlinear systems except the simplest cases, and to multiple input, multiple output systems. Thus classical techniques are not amenable generally to the design of optimal or adaptive controllers.

With the advent and abundant utilization of digital computers, computer-aided control system design has become a popular research topic. However, most contemporists abandoned the classical techniques and explored the realm of optimal control theory delving in systems described by state variables, i.e., a set of first order differential equations with design objectives described in a cost functional. Although many enlightening methods and results have been evidenced, the weaknesses are inherent. For example, the optimal control law is extremely dependent upon the proposed cost function and in many cases the correct cost function debatable [1]*. Furthermore, all states are assumed available for feedback. Even with observer theory to reproduce unmeasured variables, and subsequently allow fewer measurements, the computer storage required is excessive.

The current impression, that in order to utilize computer facilities the control problem must be implemented in state variable form, must be resolved. There is, generally speaking, no substitute for state variable techniques when applied to simple control

*Numbers in square brackets designate referenced items.

systems, but when solving relatively large plants the handling and storing of state matrices can be unthinkable. Thus if the classical theory can be extended to include modern concepts, as well as computer techniques, then possibly a more effective control tool will evolve. An interesting aspect of this approach is that it clearly shows the return to prominence of the classical frequency domain techniques in modern system analysis.

Literature Survey of Previous Work

In the past, several papers have appeared discussing various approaches to computerized classical design of control systems. Generally, performance specifications are satisfied by frequency response and root locus methods using trial-and-error procedures. According to [2], there are three digital techniques which appear to have merit in this regard: Automatic Frequency Domain Synthesis of Multiloop Control Systems (AUTO), Compensator Improvement Program (CIP), and Computerized Optimization of Elastic Booster Autopilots (COEBRA). In addition to these, developments by Nail on the Eigenvalue Encouragement Technique [3], as well as Mancini's Computer Aided Control System Design Using Frequency Domain Specifications (CALICO) [4] and Vines' Computer Automated Design of Systems (CADS) [5] are also of interest. Each of these algorithms is summarized, and the disadvantages and limitations are discussed.

Automatic Frequency Domain Synthesis of Multiloop Control Systems (AUTO)

The algorithm AUTO [6], developed at the Aerospace Corporation, was designed to aid in the synthesis of compensation for multi-loop, time-invariant control systems exemplified in Figure 1. In this figure, the plant $P(s)$ is assumed to have fixed characteristics, a single control input, and multiple outputs. With the feedback path broken at α an open-loop transfer function is defined as:

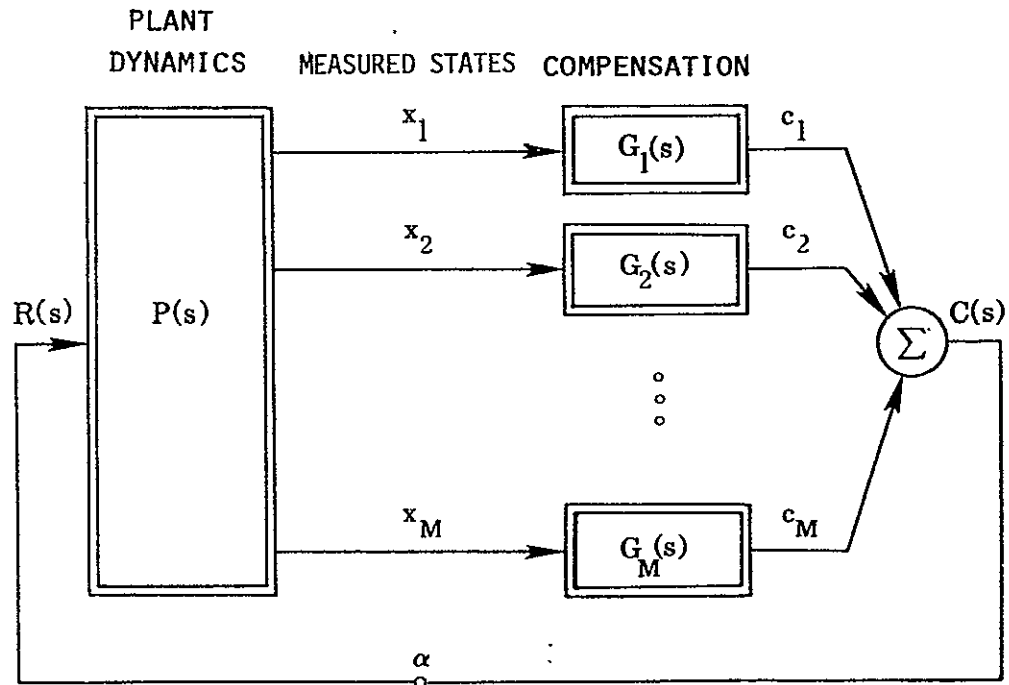
$$c(s) = C(s)/R(s) \quad (1.1)$$

assuming $R(s)$ is unity for all frequencies $s = j\omega$, then $C(j\omega)$ is the open-loop frequency response.

The philosophy of AUTO is to fit the open-loop frequency response $C(j\omega)$ to the desired open-loop response $\hat{C}(j\omega)$ by selection of the parameters of the compensators, $G_1(s)$, $G_2(s)$, ..., $G_M(s)$. The compensators are varied algebraically by making incremental changes in their parameters. Each compensator is assumed to be a rational function of the form:

$$G_k(s) = \frac{\sum_{i=1}^M a_{ki}(s)^{i-1}}{[1 + \sum_{i=2}^N b_{ki}(s)^{i-1}]} \quad (1.2)$$

where M and N are chosen by the designer and a_{ki} , b_{ki} represent the compensator coefficients. The number of compensators is dictated by the number of plant outputs.



$R(s)$ IS THE SCALAR INPUT RESPONSE

$C(s)$ IS THE OUTPUT RESPONSE

$P(s)$ IS THE $M \times 1$ MATRIX TRANSFER FUNCTION DESCRIBING THE PLANT

$x_j(s)$ REPRESENTS THE MEASURED STATE OF THE (J)TH CHANNEL;

$J = 1, 2, \dots, M$

$G_j(s)$ REPRESENTS THE (J)TH CHANNEL COMPENSATION; $J = 1, 2, \dots, M$

Figure 1. General Configuration for a Single Control Input System.

The measurement of closeness J between the actual response $C(s)$ and desired response $\hat{C}(s)$ is the mean square difference at a set of selected frequency points ω_k , $k = 1, 2, \dots, K$, that is

$$J = ||(\hat{C}^* - C^*)^T \bar{W}^T \bar{W}(\hat{C} - C)|| \quad (1.3)$$

where the asterisk (*) denotes complex conjugate and

$$\hat{C}^T = [\hat{C}(j\omega_1) \quad \hat{C}(j\omega_2) \quad \dots \quad \hat{C}(j\omega_K)] \quad (1.4)$$

$\bar{W}$ is a diagonal matrix of the form

$$\bar{W} = \begin{bmatrix} W & 0 & \dots & 0 \\ 0 & W & \dots & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & W_K \end{bmatrix} \quad (1.5)$$

that is used to assign different weights to the errors of different frequency points. The compensators are designed by varying their coefficients so that J is minimized by a gradient search method.

The directional vector along which the search is made is the gradient of J with respect to percentage changes in compensator coefficients; this type of directional vector is used to avoid premature convergence from encountering steep valleys or ridges associated with the function J .

Compensator Improvement Program (CIP)

CIP is a computerized design algorithm for aiding in the compensation synthesis for multiloop, time-invariant control systems

of the form of Figure 1. The basis of this algorithm is that with the feedback loop broken at α , the compensation design is accomplished by satisfying certain frequency response specifications on the open-loop response $C(j\omega)/R(j\omega)$. CIP design specifications include the capability of obtaining gain margins (GM), phase margins (PM), stability margins (SM), and attenuation margins (AM). Both the gain and phase margins have the normal definitions except their measurements in CIP are converted to distances from the $(-1 + j0)$ point in the $GH(j\omega)$ plane. The stability and attenuation margins are defined as follows: [7]

Definition 1

For a closed-loop stable system whose open-loop frequency response is described by $GH(j\omega)$, a stability margin (SM) is defined as a relative minima of the real function,

$$|1.0 + GH(j\omega)|$$

Definition 2

An attenuation margin (AM) of the $GH(j\omega)$ frequency response for a band of frequencies such that $\omega_1 < \omega < \omega_2$ is defined as a relative maxima of the real function, $|GH(j\omega)|^2$, when $\omega \in (\omega_1, \omega_2)$.

Gain, phase, and stability margins establish desirable amounts of phase stabilization; whereas, the attenuation margin is used to insure proper amounts of gain stabilization. CIP was developed with the objective of improving the frequency response from iteration to iteration. Two possible modes of operation are available: the Sum Improved Frequency Response (SIFR), and the Total Improved Frequency Response (TIFR). The user must select one of these modes to control

the amount of the incremental changes made in the compensator parameters in initiating the program. Generally the SIFR mode allows coarser changes than the TIFR mode.

CIP employs the mathematical programming tool the Constraint Improvement Algorithm (CIA). This algorithm possesses the unique capability of producing a directional change vector for the compensator coefficients that insures the existence of a Total and/or Sum Improved Frequency Response.

Computerized Optimization of Elastic Booster Autopilots (COEBRA)

COEBRA design is achieved by solving a sequence of constrained optimization problems by minimizing a cost function. The cost function, in terms of frequency response of time domain specifications, is subject to a set of inequality constraints. The frequency response specifications include the classical phase and gain margins; whereas, the angle of attack is included in the time domain specifications.

COEBRA employs the linear programming tool, the Simplex Algorithm, to obtain a solution; whereas, the design problem for which COEBRA was developed is nonlinear in nature. However, if the cost and constraint function are approximated by a truncated Taylor series expansion, the problem becomes a linear one and a solution is obtained through a parametric programming procedure. In obtaining the truncated Taylor series expansion a finite difference technique yields the necessary partial derivatives.

Eigenvalue Encouragement Technique (POP)

POP, a numerical technique for the iterative design of linear, time-invariant control systems, attempts to design dynamic feedback compensation by affecting the closed-loop eigenvalues in a desirable manner. This technique encourages the eigenvalues to migrate either toward or further into the left half plane, or toward other specified values. This encouragement process is accomplished by solving an unconstrained minimization problem with a selected cost function.

The algorithm is based on Danilevskii's method of generating a characteristic polynomial and the assumption that the compensation is dynamic feedback. A unique relationship between a determinant and the partial derivative operation is applied to the system characteristic $|\lambda I - A|$; the result is the partial of the closed-loop eigenvalues with respect to parameters in terms of $2p(n+1)$ determinants where n is the order of the plant and compensation A matrix, p the compensator order, and λ the eigenvalue. By determinant manipulations the necessary $2p$ determinants are evaluated by Danilevskii's methods yielding results for all the eigenvalues.

Computer Aided Control System Design Using Frequency Domain Specifications

CALICO, the computer-aided compensator design algorithm by Mancini, utilizes the constrained optimization method introduced by M. J. Box. This technique requires the desired open-loop frequency response be specified for discrete frequency points. The minimization routine varies the compensator parameters in such a manner

as to minimize a cost functional based on the difference between the actual and desired frequency response of the compensated system. With the desired response as input, the author incorporates the normal frequency domain specifications such as gain and phase margins into the overall cost functional each time the algorithm evaluates the frequency response of the open-loop system, and thus eliminates the need for specialized computations to determine margin satisfaction.

Computer Automated Design of Systems

The automated digital computer technique by Vines, CADS, is a control system compensator design oriented in the time domain. In order to minimize a specific cost functional and set the free system parameters, the technique requires as input the desired output response and system description. The minimization technique BOXPLX by M. J. Box is employed. To simulate the system to be optimized, the author chose commonly used transfer functions which were reduced to first order linear differential equations. The equations are programmed so that the transfer function blocks can be cascaded by data card input. Several nonlinear transfer blocks are also available. The program simulates the system with known parameters and then allows all free parameters to be fixed by the optimization routine in achieving the desired response.

A Comparison

In summary, each of these algorithms has advantages and limitations. With the exception of COEBRA and POP, these methods completely ignore the possibility of multiple inputs and their interactions with system parameters. Unfortunately, POP and COEBRA have several theoretical and computational limitations.

For example, POP is a numerical technique that attempts to minimize a cost function composed of 'soft' constraints. This method should minimize the cost function, but in a practical sense, (in terms of relative stability, etc.) the final system may not be any better than the original. Further, the practical limitation of computer storage and run time may prove the infeasibility of applying this method to large systems. In fact the run time is approximately proportional to n^3 where n is the system order including compensation, and systems above 40th order require more than 128K words of core storage on a UNIVAC-1108. Another unfortunate obstacle of this technique is the inherent problem of relating frequency domain design specifications such as phase and gain margins to closed-loop pole locations of large systems.

COEBRA minimizes a cost function subject to a set of inequality constraints; the cost function is used to optimize gain and phase margins, rise time, percent overshoot, etc., while the constraints insure that the performance measurements do not degrade from iteration to iteration. The directional vector is determined so as to minimize the cost function and not violate the constraints.

Thus, COEBRA also possesses the property that the final design will not be worse than the initial. COEBRA employs a method of finite differences to determine partial derivatives. This introduces numerical inaccuracies that jeopardize the practical utility of this program. From a user's view, COEBRA is difficult to enable; it requires a thorough knowledge of the programming techniques for a user to achieve a useful design. COEBRA also requires excessive computer core storage and run time; on the UNIVAC-1108, 66K words of storage must be available; whereas, both AUTO and CIP can be executed in less than 32K words for systems of equivalent order [2]. Perhaps it was in this regard that the author found it necessary to restrict the design to no more than eight bending and/or slosh modes.

AUTO assumes a single control input plant described by frequency response information in the form of complex numbers. Although AUTO appears easy to use, the judicious selection of weighting constants for every frequency component is a designer's nightmare; in some instances, no choice of constants will yield the desired design specifications. Like POP, AUTO seeks to minimize a cost function composed of soft constraints by a gradient optimization technique; the weighted mean square difference between the actual and desired frequency response indicates the measure of closeness to the desired results.

CADS, as a time domain method, accepts only first order differential equations in describing the compensation. The computer time and storage are a function of the system order and the search area on the upper and lower bounds of the system parameters. The

routine requires a good guess on the original parameters in order to obtain a workable solution. According to Vines, the optimization routine BOXPLX may continue to search for a reduced cost functional to within some significant digit even though a practical solution already has been determined.

CALICO, although frequency domain oriented, is similar to CADS in its applicability to single input-output systems and its use of the BOXPLX optimization routine. Again the technique is cursed with an optimization method that becomes more inefficient and time consuming as the system parameters increase.

CIP has the limitation of applicability to a single input, multiple output system. In addition, CIP requires much data in the form of frequency response information. Unlike the aforementioned method, however, CIP is not an optimization technique and does not attempt to maximize or minimize a cost function; rather, CIP searches for a 'suboptimal' feasible solution by satisfying a set of strict constraints that measure the performance of a design. The computer run time is proportional to the number of frequency points used to describe the plant. If properly used, CIP results in a final design that will be better than the initial.

CIP has proven its merit in the service of space vehicle control according to NASA contract reports [7,8]. If CIP could be extended to handle designs for plants with multiple inputs and multiple outputs, it obviously would be superior to any of the aforementioned methods. Further, the inclusion of a multiple input,

multiple output capability would improve greatly the utility of this technique as a design aid.

The Research Area

The objective of this research is to achieve compensation for a multiple input, multiple output control system by developing an algorithm to facilitate fast and practical compensator design with maximum computer economy while minimizing designer effort. The relative stability method of the Constraint Improvement Algorithm [7] by McDaniel and Mitchell has been chosen to determine system performance specifications by frequency domain techniques. The work presented in this exposition develops the theory and associated modifications necessary to extend the CIP type algorithm to the multivariable control system. With permission of the author of the original CIP [7] this program will now be known mnemotechnically as CIP, since much of the philosophy and many of the techniques of the original algorithm have been retained and extended.

CHAPTER II

DESCRIPTION OF THE DESIGN ALGORITHM

An Overview of the Design Problem

Figure 2 is a schematic representation of a multivariable, linear, time-invariant feedback control system. This multivariable system may be viewed as n coupled feedback systems--one for each element of the input vector. The loop transfer function for the k th system is obtained by opening the feedback path at α_k , and then determining the response $C_k(s)/R_k(s)$ with all other input R 's set to zero.

With this view of the multivariable feedback system, the designer is faced with the problem of synthesizing controllers of n interacting systems. Using classical feedback theory a controller may be designed so that the open-loop frequency response satisfies a set of frequency response design objectives. In theory this approach easily is extended to the multivariable system. However, in this case simultaneous designs of the controllers must be made so that the n open-loop frequency responses satisfy n sets of design objectives. The simultaneity of the designs is required because of the implicit functional relationship between the design objectives of the individual systems; e.g., a controller may affect the open-loop frequency response of one system in a desirable manner, while adversely affecting the response of another system.

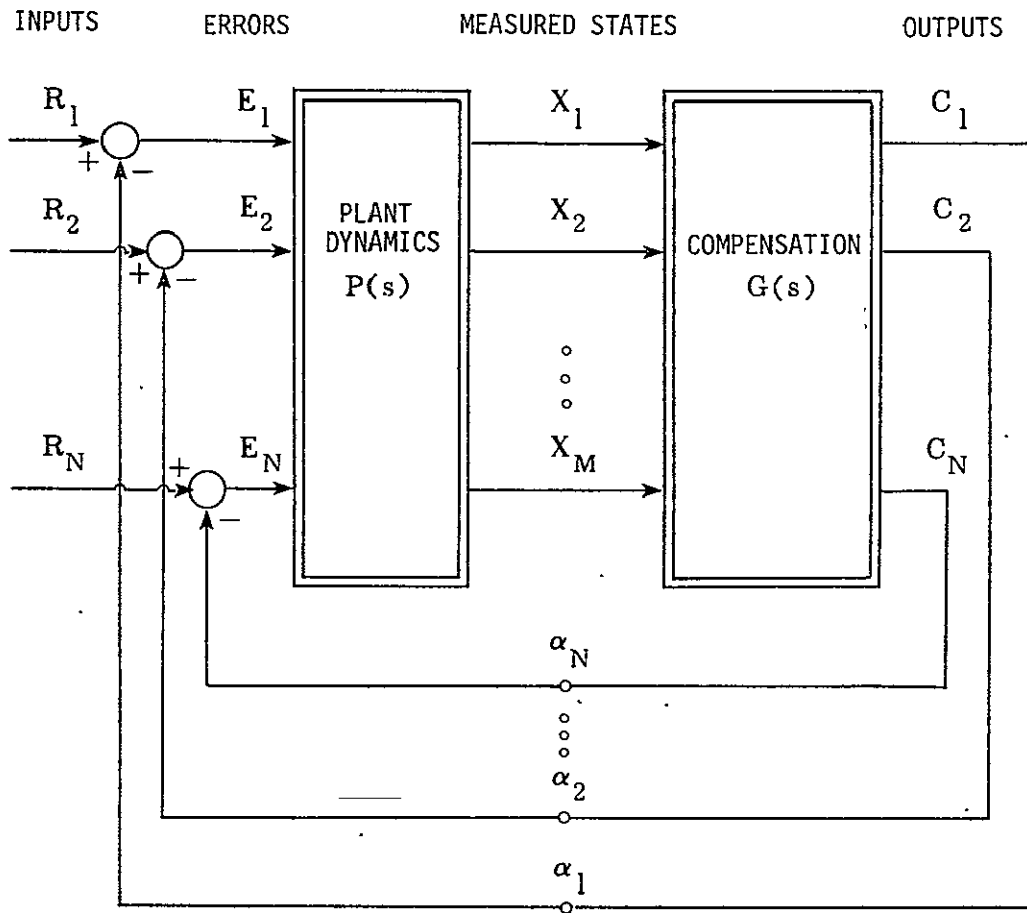


Figure 2. A Multivariable Control System.

Using classical frequency response techniques, the design of a control system to satisfy a few objectives can be accomplished with manual calculations. However, as the complexity of the system and the number of design objectives increase the development of the controller requires the aid of a high-speed digital computer.

In order to use efficiently the digital computer in a design capacity it is necessary to have a design algorithm that is amenable to digital computation; in general, such algorithms are iterative in nature. The Compensation Improvement Program [7] is an algorithm of this type that has been developed to facilitate in the design of controllers for the class of systems of Figure 1. With the loop broken at α , the algorithm determines the controllers $G_j(s)$, where j is the controller index ($j = 1, 2, \dots, M$), so that the open-loop frequency response, $C(s)/R(s)$, satisfies specified requirements. In this study a design algorithm is developed to facilitate the design of controllers for multivariable feedback systems.

The Algorithm Design Philosophy

In order to accomplish logically the design algorithm the computational flow diagram of Figure 3 has been developed. The description of the multivariable configuration requires discrete frequency data from each input to each output; whereas, the initial compensation for each controller input is described by a matrix of transfer functions. With this information the open-loop frequency response is obtained for each of the n coupled systems by determining the associated subsystem response with one loop open at a

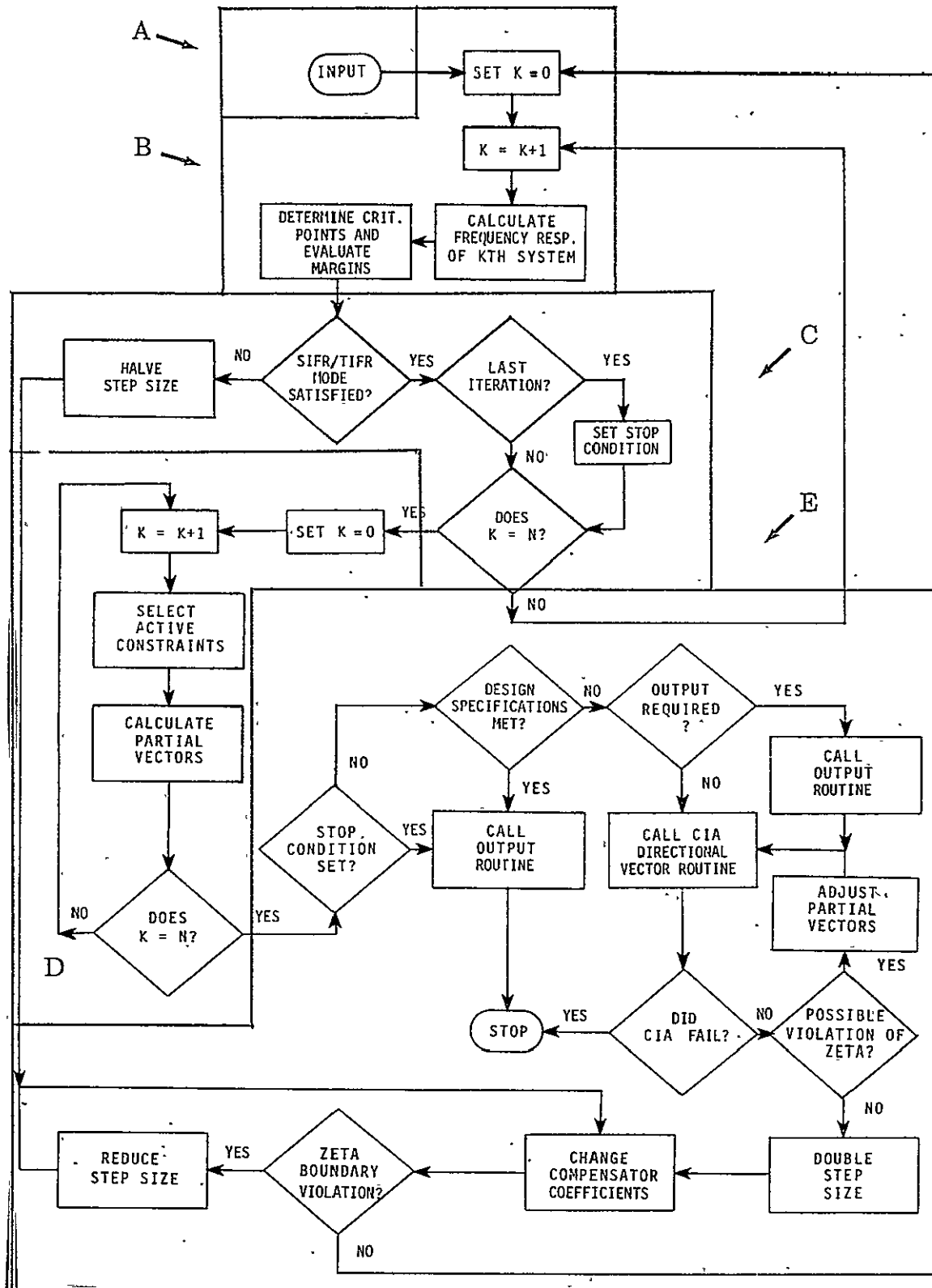


Figure 3. Computational Flow Diagram of the CIP Algorithm.

time. Likewise, for each subsystem, a set of critical points, that is; frequencies at which margins of stability or attenuation occur, is determined. It is possible to demand any number of margin requirements for gain, phase, stability, and attenuation radii. Further, these margins can be manipulated so as to make the constraints or specifications frequency dependent. An active list of radii requirements is then prepared to alleviate any margins already satisfied. From the open-loop frequency response data and the compensator coefficients, the gradient vectors of the active margins for each subsystem are calculated. Using these gradient vectors, a directional vector that can yield improvements in all active margins is determined. The free compensator coefficients are varied then in accordance with this directional vector. From this design the total response is checked to determine any margin radii not satisfied, and the process continues in an iterative manner until all specifications are met or user control, such as maximum computer time or iterations, forces a stop.

Theoretical Concepts Associated with the Algorithm

Calculation of Compensated Open-loop Frequency Response

In order to develop a CIP type algorithm for designing the controller for the multivariable system, it is necessary to have equations for calculating the open-loop frequency response for each subsystem and equations for calculating the change in each objective function (performance measurements) with respect to variations in the free parameters of the compensation. First attention is

focused on the calculation of open-loop frequency response information. From Figure 4 the output frequency response $[C(s)]$ is obtainable:

$$[C(s)] = [G(s)][P(s)][E(s)] \quad (2.1)$$

where the error or actuating function $[E(s)]$ yields

$$[E(s)] = [R(s)] - [H(s)][C(s)] \quad ; \quad (2.2)$$

the notation is defined in the Symbols table. Substituting equation (2.2) into (2.1) and solving for $[C(s)]$ yields the output relation,

$$[C(s)] = [G(s)][P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] \quad (2.3)$$

Equation (2.3) gives the closed-loop output response in terms of the input vector. Suppose that the k th diagonal element of $[H(s)]$ is set to zero and all the elements of $[R(s)]$ are nulled except the k th element which is set to unity; the result is the frequency response between the k th input and the outputs when the k th loop is open. To simplify notation, define

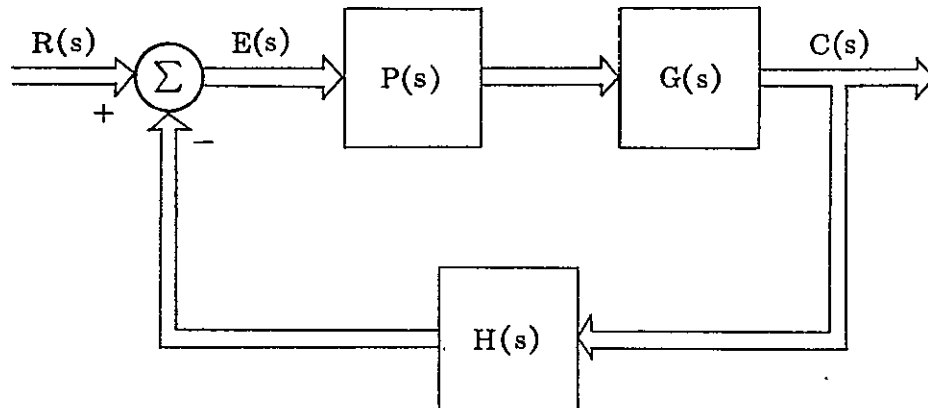
$$[V(s)] \triangleq \{I + [H(s)][G(s)][P(s)]\}^{-1} \quad (2.4)$$

and

$$[U(s)] \triangleq [G(s)][P(s)] \quad (2.5)$$

Hence the open-loop complex frequency response of the k th system is

$$\frac{C_k(s)}{R_k(s)} = \underline{u}^k(s) \cdot \underline{v}_k(s) \quad (2.6)$$



$R(s)$ IS THE $N \times 1$ INPUT VECTOR
 $E(s)$ IS THE $N \times 1$ ERROR VECTOR
 $C(s)$ IS THE $N \times 1$ OUTPUT VECTOR
 $P(s)$ IS THE $M \times N$ MATRIX TRANSFER
 FUNCTION DESCRIBING THE PLANT
 $G(s)$ IS THE $N \times M$ MATRIX TRANSFER
 FUNCTION DESCRIBING THE CONTROL
 LAW AND CASCADED COMPENSATION
 $H(s)$ IS AN $N \times N$ UNITY MATRIX

Figure 4. Vector Representation of the Multivariable Control System.

where $\underline{u}^k(s)$ and $\underline{v}_k(s)$ are, respectively, the k th row of $[U(s)]$ and the k th column of $[V(s)]$ with the k th diagonal element of $[H(s)]$ set to zero. By fixing the proper diagonal element of $[H(s)]$ to zero, it is obvious how equations (2.4), (2.5), and (2.6) can be used to evaluate the open-loop frequency response for each k th system.

Evaluation of the Critical Frequencies

Next attention is focused on the determination of the critical frequencies with respect to the design objectives. In CIP the design is accomplished by requiring the open-loop frequency response to satisfy certain specifications. These design specifications are converted to distances between certain critical points of the open-loop frequency response and certain points in the corresponding complex plane where $s = j\omega$. The typical objective function for the k th open-loop system is

$$d = \left\{ [A + C_k(j\omega)][A + C_k(j\omega)]^* \right\}^{\frac{1}{2}} \quad (2.7)$$

where A is the point in the complex plane from which the specification is measured; e.g., for stability margins A is the $(-1 + j0)$ point. In equation (2.7), the response $C_k(j\omega)$ is calculated from (2.6) with $R_k(j\omega)$ set equal to unity.

Calculation of the Partial Vectors

Now attention is directed to the calculation of the change in the design objectives with respect to the free parameters of the

controller. With this objective in mind, the partial derivative of the distance d with respect to some parameter w is

$$\frac{\partial d}{\partial w} = \frac{1}{2} \left\{ [A + C_k(j\omega)] \frac{\partial C_k^*(j\omega)}{\partial w} + \frac{\partial C_k(j\omega)}{\partial w} [A + C_k(j\omega)]^* \right\} \div d . \quad (2.8)$$

Equation (2.8) can be rewritten as

$$\frac{\partial d}{\partial w} = \operatorname{Re} \left\{ [A + C_k(j\omega)]^* \frac{\partial C_k(j\omega)}{\partial w} \right\} \div d . \quad (2.9)$$

Evaluation of (2.9) depends on determining accurately the partial term, $\partial C_k(j\omega)/\partial w$. Using the chain rule this becomes

$$\frac{\partial C_k(j\omega)}{\partial w} = \frac{\partial C_k(j\omega)}{\partial G_{ij}} \cdot \frac{\partial G_{ij}}{\partial w} \quad (2.10)$$

where G_{ij} is the element of the controller $[G(s)]$ in which the free parameter w appears with $i = 1, 2, \dots, N$ system outputs and $j = 1, 2, \dots, M$ system states sensed.

The necessary equations for evaluating the first term in (2.10) are derived in the sequel. The partial of (2.3) with respect to the controller element G_{ij} gives

$$\begin{aligned} \frac{\partial [C(s)]}{\partial G_{ij}} &= -[G(s)][P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[H(s)] \cdot \\ &\quad \frac{\partial [G(s)]}{\partial G_{ij}} [P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] + \\ &\quad \frac{\partial [G(s)]}{\partial G_{ij}} [P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] . \end{aligned} \quad (2.11)$$

Then, the $\partial C_k(s)/\partial G_{ij}$ is the k th element of (2.11) with the k th

diagonal element of $[H(s)]$ set to zero, and all the elements of $[R(s)]$ set to zero except the k th which is set to unity. In (2.11) the partial term $\partial[G(s)]/\partial G_{ij}$ is a zero matrix except for the (ij) th element which is unity.

Next consideration is given to the evaluation of the second term in equation (2.10). It is assumed that the (ij) th element of $[G(s)]$ is composed of a cascaded arrangement of transfer functions, i.e.,

$$G_{ij}(s) = \prod_{k=1}^K G_{ijk}(s) \quad (2.12)$$

where K is the number of cascaded elements. The ℓ th cascaded element of the (ij) th element of $[G(s)]$ has the general form

$$G_{ij\ell}(s) = \frac{\sum_{m=0}^M x_{ij\ell m} s^m}{\sum_{n=0}^N y_{ij\ell n} s^n} \quad (2.13)$$

Then, the free parameters of this element are the x 's and y 's. If w in (2.10) is the p th numerator coefficient of (2.13), then

$$\frac{\partial G_{ij}(s)}{\partial x_{ij\ell p}} = \prod_{\substack{k=1 \\ k \neq \ell}}^K G_{ijk}(s) \frac{+s^p}{\sum_{n=0}^N y_{ij\ell n} s^n} \quad (2.14)$$

or

$$\frac{\partial G_{ij}(s)}{\partial x_{ij\ell p}} = G_{ij}(s) \frac{+s^p}{\sum_{m=0}^M x_{ij\ell m} s^m} \quad (2.15)$$

Similarly if w in (2.10) is the p th denominator coefficient of (2.13), then

$$\frac{\partial G_{ij}(s)}{\partial y_{ijlp}} = \prod_{\substack{k=1 \\ k \neq l}}^K G_{ijk}(s) \frac{-s^p}{\left(\sum_{n=0}^N y_{ijln} s^n \right)^2} \quad (2.16)$$

or

$$\frac{\partial G_{ij}(s)}{\partial y_{ijlp}} = G_{ij}(s) \frac{-s^p}{\left(\sum_{n=0}^N y_{ijln} s^n \right)} \quad (2.17)$$

By appropriately using equations (2.3), (2.9), (2.10), (2.12), (2.15) and (2.17), the first order change of any CIP objective function with respect to the free parameters of the controller can be calculated.

CHAPTER III

DESCRIPTION OF THE COMPUTATIONAL FLOW DIAGRAM

Implementation of the Algorithm

The algorithm implementation evolved with the following objectives in mind.

1. Any linear, time-invariant system structure should be acceptable.
2. Numerical problems should not restrict the method to low order systems.
3. Computational requirements should be reasonable for high order systems.
4. Monitoring of compensation at desired iteration levels should be possible.
5. Partial derivatives should be exact.

A Synopsis of the Algorithm

Keying on the aforementioned goals, possibilities were weighed to determine the most effective methods of implementing and computer coding the algorithm. The following is a description of the principles and computational logic involved in producing an executable version of the compensator improvement program for the multivariable control system of Figure 2. In this sequel the logic flow diagram

of the Continuance Criterion, (D) The Determination of the Gradient Vectors Corresponding to the Active Constraints, and (E) Calculation of the Directional Vector and the Compensation Enhancement. For a detailed explanation of the particular subroutines used, refer to alphabetical listing in Appendix A. The Fortran IV computer code is listed alphabetically in Appendix B.

Section A: Data Input

Referring to the schematic diagram of Figure 2, recall that the multivariable system may be viewed as n coupled feedback systems--one for each element of the input vector. With this view, the loop transfer function of the k th system is obtainable by opening the feedback path at α_k and determining the response $C_k(s)/R_k(s)$ with all input R 's set to zero except the k th element. Thus the variable k in the computational flow diagram of Figure 3 is defined as the respective subsystem in accordance with the corresponding element of the input vector.

The input description of the multivariable configuration requires discrete frequency data from each input to each output in describing the plant system; whereas, the initial compensation for each controller is described by a matrix of transfer functions. With this information the open-loop frequency response is obtained for each of the n coupled systems. Likewise, for each subsystem, a set of critical points, that is, frequencies at which margins of stability or attenuation occur, is determined. Hence, the input routine requires data of four types as shown in Table 1 and clarified in the following discussion.

Table 1. Outline of CIP Data

1. Iteration Control
 - a. Mode, identification code
 - b. Start, stop, print iterations
 - c. Maximum, minimum step sizes
 - Etc.
2. Design Specifications
 - a. Desired Stability and Attenuation Radii
 - b. Frequency Ranges over which searches for critical points are to be made
3. Description of Plant
 - a. Number of Control Inputs
 - b. Number of Outputs
 - c. Discrete frequency response data
4. Description of Compensation
 - a. Gain Constant in each channel
 - b. Number of Subcompensators in each channel
 - c. Coefficients for each subcompensator in first and second order factors only
 - d. Constraints to be placed on the coefficients

1. Iteration Control

First, user control parameters are entered; these include the extremum step sizes to be taken on iterations, maximum iterations for convergence, designation of iterations to be printed, user identification code, etc. Here also the user must specify the mode used in the program to determine when an iteration has been completed. In particular, the mode designates which continuance criterion must be used to determine whether the trial design at the $(i + 1)$ th iteration is an improvement in comparison to the results at the i th iteration. One of two modes must be chosen:

- i. Total Improved Frequency Response Mode (TIFR) requires that from iteration to iteration no unsatisfied objectives or design specifications are allowed to degrade and insures improvement in at least one.
- ii. Sum Improved Frequency Response Mode (SIFR) requires that the sum improvement exceed the sum degradation from iteration to iteration.

It is obvious that the TIFR mode produces a more stringent continuance criterion on the compensation.

2. Design Specifications

The second portion of the input data designates the design specifications for achieving relative stability and relative attenuation. In particular, the mathematical formulation of the design problem is to determine the free parameters of the compensators such that the objective functions satisfy a set of design specifications, i.e., gain, phase, stability and attenuation

margins. Mathematically, if a total of n critical frequency response points are chosen, the problem can be expressed as a strict constraint mathematical programming problem of the form:

Determine the $\underline{x}^T$

such that the constraints $g_i(\underline{x}^T) \geq b_i$, $i = 1, 2, \dots, n$; (3.1)

where $\underline{x}$ represents the free compensator parameters, b_i represents the design specifications, and $g_i(\underline{x}^T)$ contains the objective functions, that is, frequency response limitations and constraints. Thus the general idea is to change the compensator coefficients so that each constraint comes closer to being satisfied at each iteration. Other methods of obtaining the design objectives could be implemented; however, from a practical point of view the method of the strict constraint problem is particularly appealing in producing a change vector for the compensator coefficients that insures the existence of a Total and/or Sum Improved Response. Furthermore, this method allows the margin radii specifications to become frequency dependent. Conceivably, it is desirable that regions of the frequency response be various distances from the $(-1 + j0)$ point in the $GH(j\omega)$ plane while other regions be constrained to be greater or less than limitations with respect to the origin of the $GH(j\omega)$ plane. Thus in general a frequency response is desired to have some basic shape which can be translated with respect to frequency and not constrained to match exactly a desired frequency response.

3. Description of Plant

The multivariable configuration of Figure 2 may be described in two parts, the plant and the controller. First the description of the uncompensated plant requires discrete frequency response data between each input and each output channel. The choice of the discrete data description for the plant was made to avoid computational difficulties that might be encountered in evaluating high order transfer functions and to conserve computing time in the iterative process of CIP. Furthermore, discrete frequency response data is often the best information available for describing the system.

4. Description of Compensation

Secondly, the initial compensation or controller design is necessary for each control input. Again recall the objective of this work is not to develop a self-contained, computer-aided design algorithm but to provide the control engineer with a design aid. In this regard, it is assumed that the designer knows the control law necessary to enhance his design objectives. Normally the engineer uses s-domain rational functions in investigating designs, thus for simplicity, the compensation elements are described by transfer functions in the form of cascaded first and second order factors, that is,

$$G(s) = (\text{GAIN}) \frac{\prod_{i=1}^{N1} (ZA_i + ZB_i s) \prod_{j=1}^{N2} (ZC_j + ZD_j s + ZE_j s^2)}{\prod_{i=1}^{M1} (PA_i + PB_i s) \prod_{j=1}^{M2} (PC_j + PD_j s + PE_j s^2)} \quad (3.2)$$

Section B: Frequency Response Manipulations

Figure 5 gives a detailed expansion of the logic of Section B in Figure 3. Note that Section B is composed of four major routines in determining the frequency response: Delete Points, Add Points, Calculation of the Closed-Loop Frequency Response, Calculation of Critical Points.

The Delete Points routine is designed to remove any frequency points and corresponding response terms no longer of major concern which might have been added for accuracy on previous iterations; however, the original data are always retained. The routine is designed to save computer storage as well as computer time in response calculations and in scanning for margins. This algorithm is coded in the subprogram DELETE [10].

The Calculation of the Closed-Loop Frequency Response is performed by implementing the equations developed in Section 3 of Chapter II. In particular, the closed-loop output vector $[C(s)]$ is determined by the relation,

$$[C(s)] = [G(s)][P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] \quad (3.3)$$

Equation (3.3) gives the closed-loop output vector in terms of the input vector. Based on the theoretical concepts of Chapter II, suppose that the k th diagonal element of $[H(s)]$ is set to zero and all the elements of $[R(s)]$ are nulled except the k th element which is set to unity; the result is the frequency response between the k th input and the outputs when the k th loop is open. Utilizing the aforementioned notation,

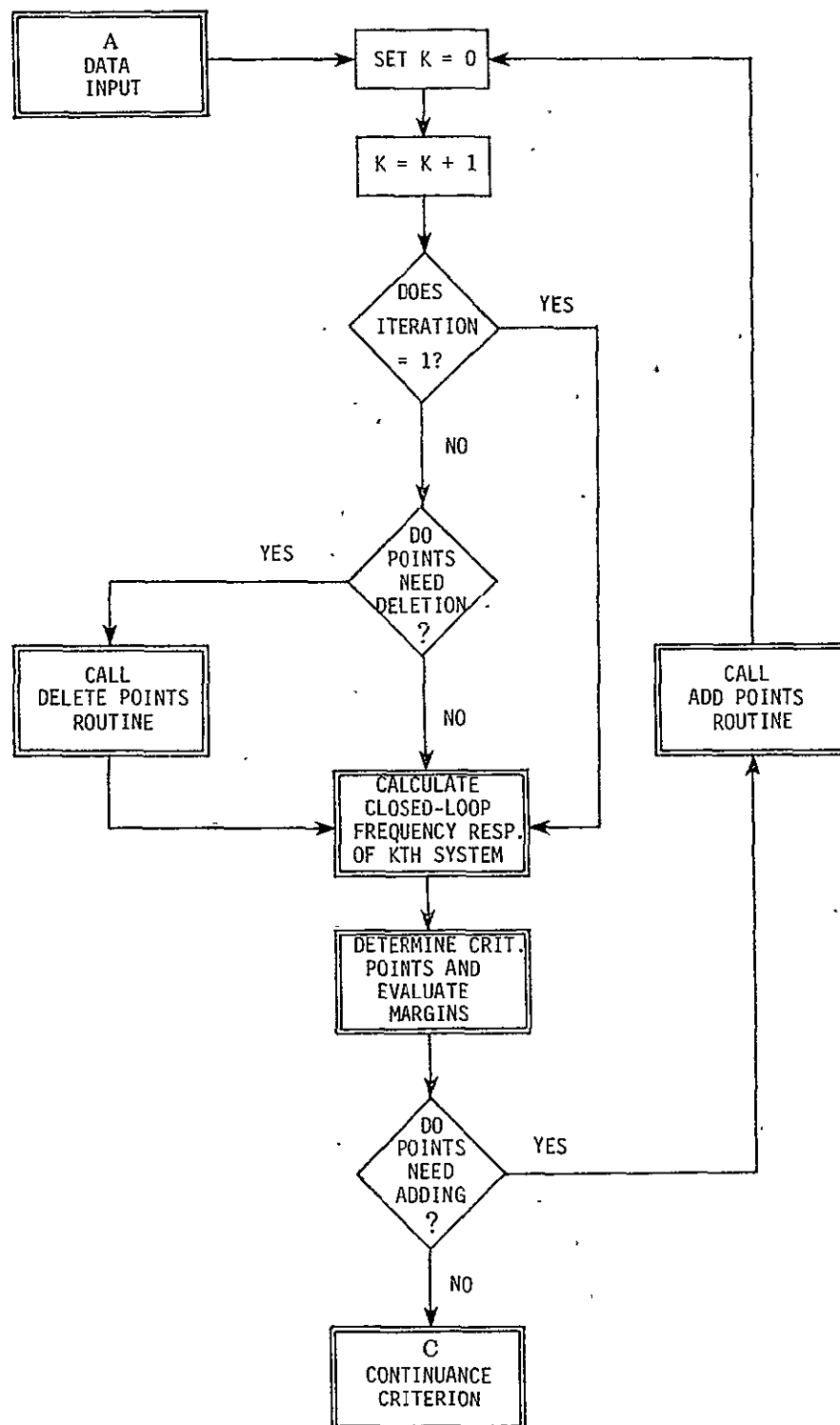


Figure 5. Logic Diagram Representing the Frequency Response Manipulations of Section B.

$$[V(s)] \triangleq \{I + [H(s)][G(s)][P(s)]\}^{-1} \quad (3.4)$$

and

$$[U(s)] \triangleq [G(s)][P(s)] \quad . \quad (3.5)$$

The open-loop frequency response of the k th system is

$$\frac{C_k(s)}{R_k(s)} = \underline{u}^k(s) \underline{v}_k(s) \quad , \quad (3.6)$$

where $\underline{u}^k(s)$ and $\underline{v}_k(s)$ are, respectively, the k th row of $[U(s)]$ and k th column of $[V(s)]$. By setting the proper diagonal element of $[H(s)]$ to zero, equations (3.4), (3.5), and (3.6) can be used to evaluate the frequency response of each system.

This algorithm is coded in the main program using two sub-routines: EVAL and CRT. The subprogram EVAL evaluates the controller at the specified frequencies with the aid of the program POLEV, a polynomial evaluation routine. EVAL then determines the product of the controller response and the plant response, that is, $[G(s)] \cdot [P(s)]$. With this transfer relation the subprogram CRT determines the total response $[C(s)]$ and selects the open-loop frequency response of the k th system.

The Determination of the Critical Points is designed to yield the critical points of the open-loop frequency response and to ascertain whether they satisfy certain design specifications for achieving the relative stability and relative attenuation margins. Recall these design specifications are expressed mathematically as the distances between certain critical points of the frequency

response and particular points in the corresponding complex plane. The typical objective function for the k th open-loop system is

$$d = \left\{ [A + C_k(j\omega)][A + C_k(j\omega)]^* \right\}^{\frac{1}{2}} \quad (3.7)$$

where A is the point in the complex plane from which the specification is measured; for stability, gain, and phase margins A is the $(-1 + j0)$ point; for attenuation margins, the $(0 + j0)$ point is chosen. The design specifications include subprograms for determining the gain, phase, stability, and attenuation margins.

The subprogram to Add Points is designed to provide more data around each of the critical frequency points, thereby, yielding a more exact margin value without the input of excessive data and the consequent increase in storage. This algorithm is encoded as ADDPTS and uses an interpolate routine INTER, as well as the aforementioned EVAL and CRT routines to update the frequency response for each added data point.

As indicated in Figure 5, the frequency response and critical margins are calculated for each k th system adding and deleting frequency points as necessary.

Section C: Evaluation of the Continuance Criterion

Figure 6 represents the logic decision blocks of Section C in Figure 3. These decision blocks are encoded in the main program and are designed to force the program into the specified continuance criterion when an iteration has been completed. Recall that the continuance criterion mode must be specified by the user as input

data to determine whether the trial design at the $(i + 1)$ th iteration is an improvement in comparison to the results of the i th iteration. The two modes available, the TIFR mode and the SIFR mode, are as defined in Section A.

Note that for the first iteration the mode block is bypassed thereby assuming that an improved solution has occurred and allowing the program to continue to the determination of the partial vectors.

If the SIFR/TIFR condition for the current iteration is satisfied the program checks user control data to decide whether the maximum iteration condition has been exceeded. If the last iteration has been reached the program sets a stop condition which prohibits further manipulations of the active constraints and partial vectors. Assuming the maximum iteration code has not been met, the main program directs control to Section D.

Now if the SIFR/TIFR criterion has not been met, the program interprets this condition to mean that the change in the compensator coefficients was too large and control proceeds to decrease the step size of the previous iteration in an attempt to force an improved solution.

Section D: Determination of the Gradient Vectors

Figure 7 is an expanded view of Block D of Figure 3; Section D is concerned with the determination of the active constraints, that is, the margins which do not satisfy the required design specifications, and their relation in evaluating the partial vectors.

The selection of active constraints is coded within the main program. CIP checks to determine which of the specified stability

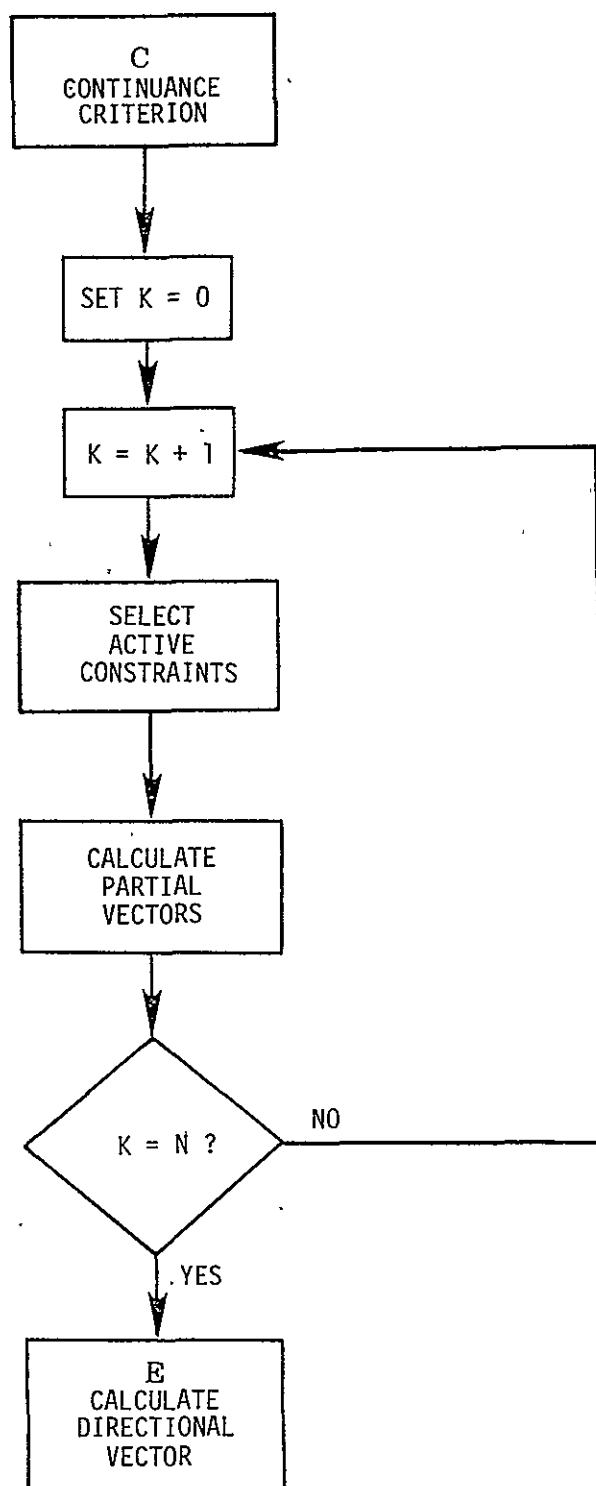


Figure 7. Logic Diagram Representing the Calculation of the Gradient Vectors of Section D.

and attenuation margins do not satisfy the margin design specifications entered by the user as input data. As previously noted, it is possible to demand any combination of margin requirements as gain, phase, stability, and attenuation radii. These margins can be manipulated so as to make the specifications or constraints frequency dependent. A list of the active radii requirements, margins, and corresponding frequencies is then prepared to alleviate any margins already satisfied. The program checks for any duplications in margins, in which case retaining only the first such critical point found.

The second objective of Section D is the calculation of the partial vectors as described in Section 3 of Chapter II. Recall that the partial vectors represent the change in the design objectives with respect to the free parameters, that is, the compensator coefficients of the controller. Thus the partial derivative of the objective function d with respect to some parameter w can be expressed as

$$\frac{\partial d}{\partial w} = \text{Re} \left\{ [A + C_k(j\omega)]^* \frac{C_k(j\omega)}{\partial w} \right\} \div d \quad (3.8)$$

Accordingly, the partial term $\partial C_k(j\omega)/\partial w$ can be expanded by the chain rule as

$$\frac{\partial C_k(j\omega)}{\partial w} = \frac{\partial C_k(j\omega)}{\partial G_{ij}} \cdot \frac{\partial G_{ij}}{\partial w} \quad (3.9)$$

where G_{ij} is the element of the controller $[G(s)]$ in which the free parameter w appears.

Evaluation of the first term in (3.9) yields the relation

$$\begin{aligned} \frac{\partial[C(s)]}{\partial G_{ij}} &= - [G(s)][P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[H(s)] \cdot \\ &\quad \frac{\partial[G(s)]}{\partial G_{ij}} [P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] + \\ &\quad \frac{\partial[G(s)]}{\partial G_{ij}} [P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] \quad (3.10) \end{aligned}$$

Then, $\partial C_k(s)/\partial G_{ij}$ is the k th element of (3.10) with the k th diagonal element of $[H(s)]$ set to zero, and all the elements of $[R(s)]$ set to zero except the k th which is set to unity. In (3.10) $\partial[G(s)]/\partial G_{ij}$ is a zero matrix except for the (ij) th element which is unity.

Evaluation of the second partial term in equation (3.9) is acquired by assuming that the (ij) th element of the compensator matrix $[G(s)]$ is composed of a cascaded arrangement of transfer functions, i.e.,

$$G_{ij}(s) = \prod_{k=1}^K G_{ijk}(s) \quad (3.11)$$

where K represents the number of cascaded elements. Thus the ℓ th cascaded element of the (ij) th compensator has the general form

$$G_{ij\ell}(s) = \frac{\sum_{m=0}^M x_{ij\ell m} s^m}{\sum_{n=0}^N y_{ij\ell n} s^n} \quad (3.12)$$

where the free parameters of this element are the x 's and y 's.

Assuming the parameter w in (3.9) is the p th numerator coefficient in (3.12), then the following expression is obtained:

$$\frac{\partial G_{ij}(s)}{\partial x_{ijlp}} = G_{ij}(s) \frac{+s^p}{\left(\sum_{m=0}^M x_{ijlm} s^m \right)} \quad (3.13)$$

Similarly, letting w represent the p th denominator coefficient in equation (3.9), then

$$\frac{\partial G_{ij}(s)}{\partial y_{ijlp}} = G_{ij}(s) \frac{-s^p}{\left(\sum_{n=0}^N y_{ijln} s^n \right)} \quad (3.14)$$

Thus equations (3.10), (3.13), and (3.14) can be used effectively to determine the first order change of any CIP objective function with respect to the free parameters of the controller.

As indicated by the decision block, the partial vector of each system is tabulated and stored in an orderly array for later use in determining the directional vector in Section E. The partial vector routine is coded in the subprogram PARTIAL in conjunction with the frequency response subroutine CRT; subroutine CRT determines the partial term $\partial C_k(j\omega)/\partial G_{ij}$ in equation (3.9).

Section E: Calculation of the Directional Vector and

Compensation Enhancement

Of major significance in Section E is the manipulation of the system partial vectors in obtaining a directional vector using the constraint improvement algorithm. (See Figure 8.)

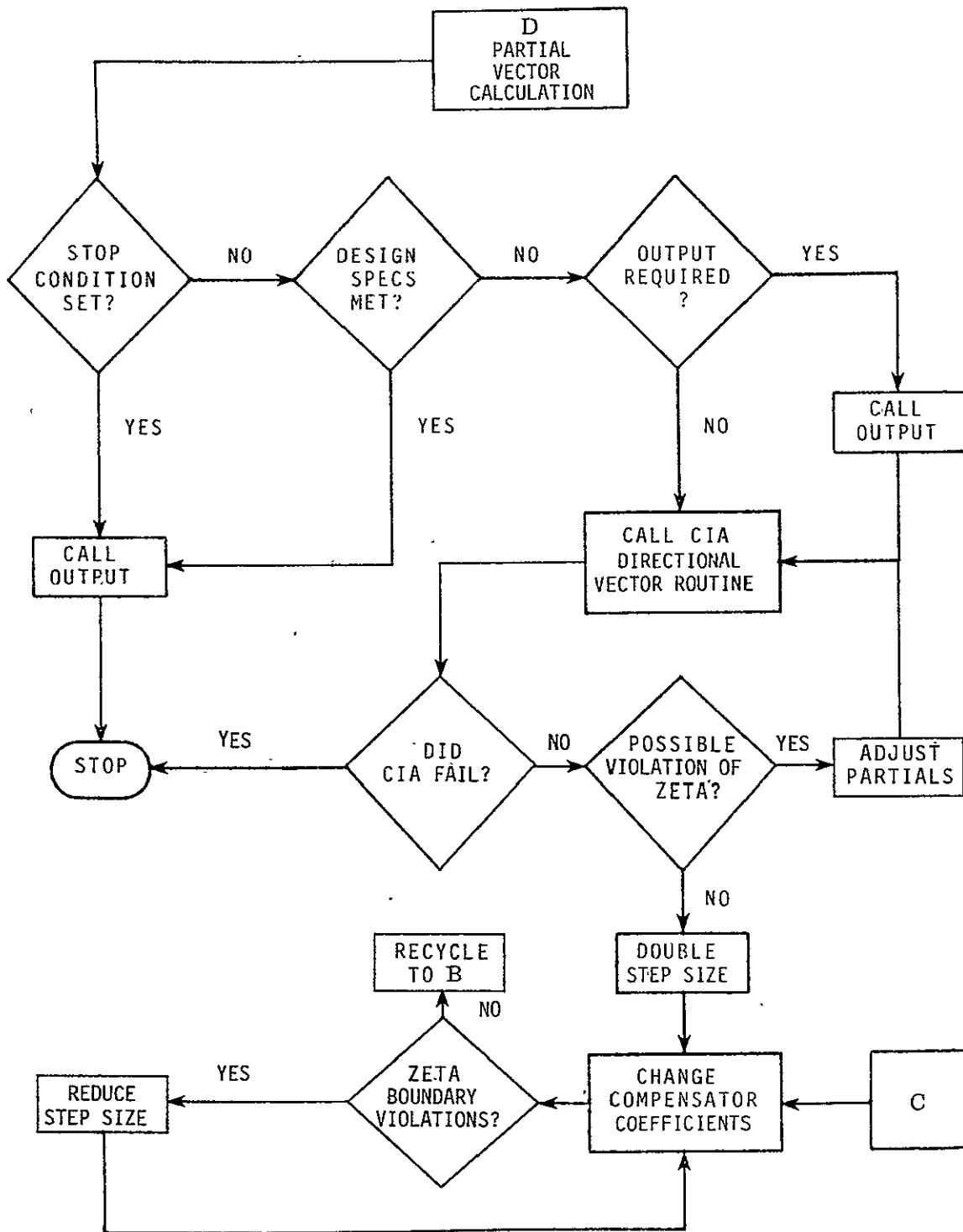


Figure 8. Logic Diagram Representing the Calculation of the Directional Vector and Corresponding Compensator Enhancement of Section E.

First the decision block of Section E tests to ascertain whether the stop condition has been set. If so, the output routine OUTPT is called; otherwise, the main program determines whether the design specifications have been met and calls the output routine if necessary. Assuming the design objectives are not satisfied, the program checks user data to determine whether an output is desired for the particular iteration. The directional vector then is calculated with the aid of the Constraint Improvement Algorithm [7] in the subprogram DIRVEC. This subprogram also checks for routine failure yielding a stop command. The subprograms MATMUL and MATINV are auxiliary routines to DIRVEC for matrix multiplication and inversion, respectively.

In Figure 8, after calculating the directional vector and assuming the CIA did not fail, CIP investigates the user option of constraining the poles and zeros of compensation to lie within a specified damping ratio sector. By limiting the compensation to first and second order factors, the complexity of constraining the compensation poles and zeros to lie within a sector defined by constant damping ratio lines as shown in Figure 9 is reduced greatly. Hence the next step is the determination of the directions of movement of the compensator poles and zeros on the specified zeta boundaries. In order to avoid a zeta violation, the directions of these poles and zeros must be along the boundaries or into the defined sector. If with the present directional vector the movements of these poles and zeros are in the wrong directions, judiciously selected terms in the partial vectors are nulled and the

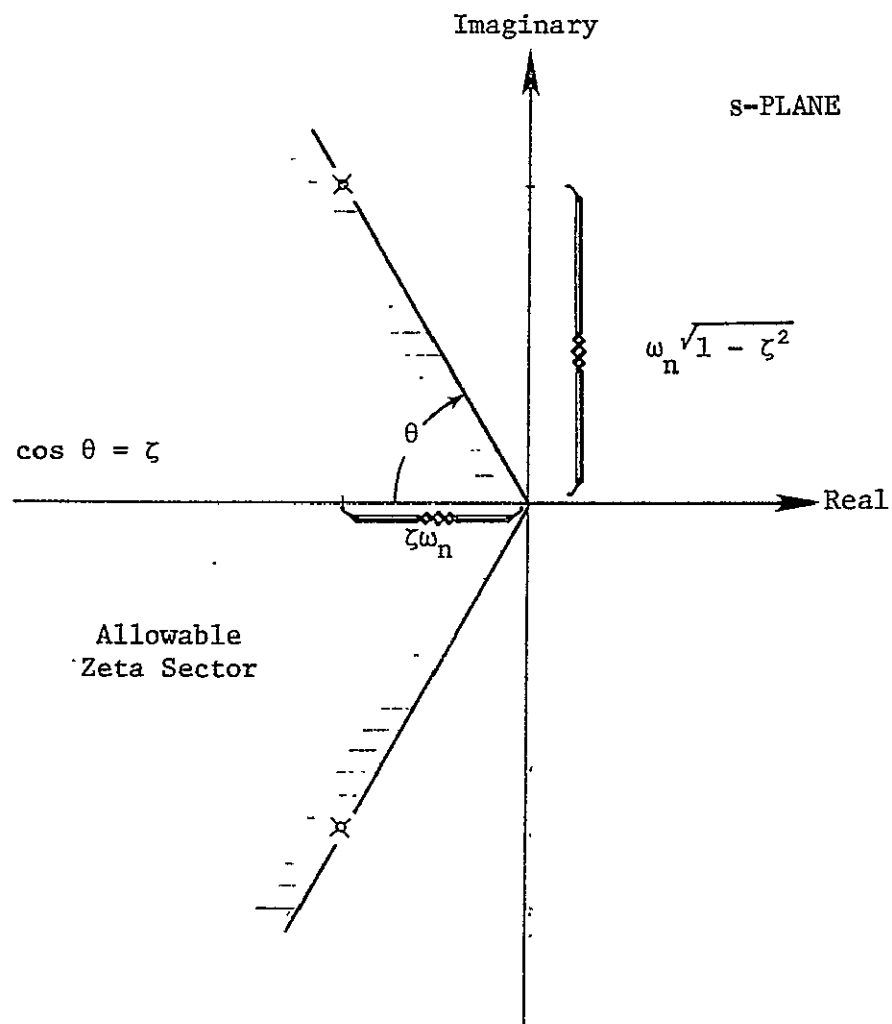


Figure 9. s-Plane Configuration of Desirable Compensation and Typical Pole Locations for a Second Order System.

directional vector is recomputed. This process is continued until the directions of movement of all compensation poles and zeros on the boundaries do not result in a zeta violation. The checking of the directions of movement of these poles and zeros and the setting of the elements of the partial vectors to zero is accomplished by the subprogram XCHECK. This routine assures the existence of a non-zero step size that will not produce a zeta violation by poles and zeros on the boundaries.

Referring to Figure 8, after an acceptable directional vector has been established in accordance with XCHECK, a step size is selected, and in conjunction with this directional vector, the individual compensator coefficients are effectively augmented. At this point, a violation in the zeta constraints can occur from an inappropriate selection of the step size, i.e., usually if the step size is too large. Thus, the zeta constraints are checked. If a violation occurs, the maximum step size that will not produce a violation is computed and the compensator coefficients are incremented; otherwise, the program recycles to Section B in Figure 3. The check for zeta violations, as well as the computation of a maximum acceptable step size, is accomplished by the subroutine YCHECK. The theory underlying this routine along with additional usage information is presented in Appendix A.

CHAPTER IV

INVESTIGATIONS AND EXAMPLES

In order to illustrate the practical utility of the multivariable Compensator Improvement Program, the improvements of the compensators for space related examples are presented. This by no means limits the scope of the work to space oriented control systems, but rather provides large system problems which have been investigated by other means. In particular, three examples are discussed: (1) a dual input, dual output system with uncoupled characteristics; (2) a dual input, dual output system with coupling; (3) a dual input, four output system exhibiting coupled characteristics.

Dual Input, Dual Output System

In this example the system under consideration is similar to that of Figure 2 with $M=2$ controller inputs or measured states, and $N=2$ controller outputs. Figure 10 shows the actual subsystem under investigation and is representative of the attitude control system for a finned launch vehicle at a specified flight time following launch [11]. Each subsystem has plant dynamics $C_k(s)/R_k(s)$ described by the uncompensated frequency response plot of Figure 11 where k represents the number of controller inputs and hence the number of subsystems; k equals two in this example. Table 2 exhibits the twenty-eight discrete frequency points chosen to describe the open-loop response of each subsystem. The compensation

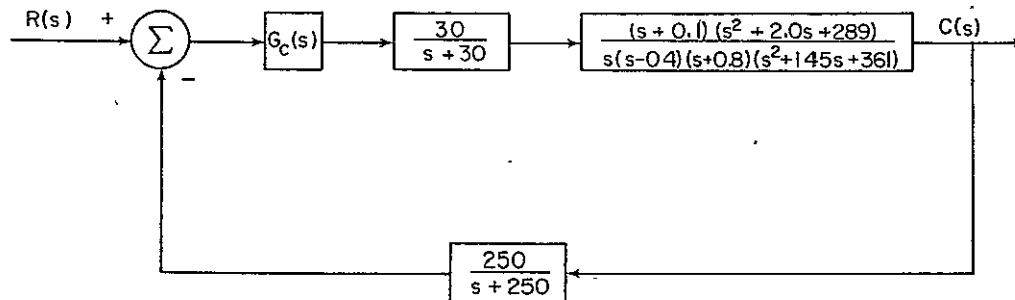


Figure 10. Block Diagram Representing Each Subsystem of the Finned Vehicle Example.

Table 2. Frequency Response Input Data Describing the Plant in the Finned Vehicle Example.

Data Points	Complex Frequency		Uncompensated Plant Response	
	RE[s]	IM[s]	RE[P]	IM[P]
1	0.000	0.100	-69.6000	54.9000
2	0.000	0.132	-67.3000	36.6000
3	0.000	0.178	-63.3000	20.8000
4	0.000	0.240	-57.2000	8.9000
5	0.000	0.347	-46.9600	- 0.5890
6	0.000	0.501	-34.8000	- 4.3200
7	0.000	0.646	-26.6000	- 4.3300
8	0.000	1.230	-10.8800	- 1.5220
9	0.000	2.000	- 4.7100	- 0.3060
10	0.100	2.800	- 2.4800	- 0.1840
11	0.200	3.600	- 1.5100	- 0.1020
12	0.250	4.000	- 1.2200	- 0.0750
13	0.250	4.921	- 0.8079	0.0018
14	0.250	6.000	- 0.5369	0.0340
15	0.200	6.400	- 0.4690	0.0469
16	0.100	7.200	- 0.3630	0.0603
17	0.000	8.000	- 0.2870	0.0645
18	0.000	10.071	- 0.1666	0.0478
19	0.000	12.400	- 0.0956	0.0309
20	0.000	15.600	- 0.0402	0.0013
21	0.000	18.327	- 0.5180	- 0.1090
22	0.000	19.191	- 0.1924	- 0.0525
23	0.000	19.638	- 0.1677	0.0249
24	0.000	20.563	- 0.0969	0.0534
25	0.000	30.415	- 0.0139	0.0182
26	0.000	40.095	- 0.0047	0.0092
27	0.000	60.686	- 0.0006	0.0031
28	0.000	100.000	- 0.0001	0.0007

matrix consists of identical compensators in the diagonal elements, i.e.,

$$G_{11}(s) = G_{22}(s) = \frac{(1 + 10s)(5 + 1.25s)}{(1 + 11s)(5 + 1.00s)} ; \quad (4.1)$$

whereas, the off-diagonal terms are chosen as zero to inhibit any coupling terms.

It is desired to modify the compensators, $G_{11}(s)$ and $G_{22}(s)$, so that the closed-loop step response of each subsystem reasonably is damped and "ringing" caused by the low-damped high frequency modes is negligible. Further, the DC gain of each compensator is chosen so that the magnitudes of the steady-state errors to a velocity input are less than 0.15. These specifications are satisfactorily achieved by requiring that

- (1) all SM's ≥ 0.5 when $0 \leq \omega \leq 16.0$,
 - (2) all AM's ≤ 0.1 when $16 \leq \omega \leq 100$,
 - and (3) the DC gain of the compensator is greater than 26.67.
- (4.2)

After 31 iterations, approximately 36 seconds of CPU time on a Univac 1108 Computer, the design compensation is obtained as

$$G_{11}(s) = G_{22}(s) = \frac{(1.0 + 1.60084s)(5.0 + 5.57314s)}{(1.0 + 16.3982s)(5.0 + 1.14051s)} . \quad (4.3)$$

The compensated frequency response for each k th subsystem is illustrated in Figure 12 in which all design specifications have been accomplished. In particular, Entry A in Table 3 shows the

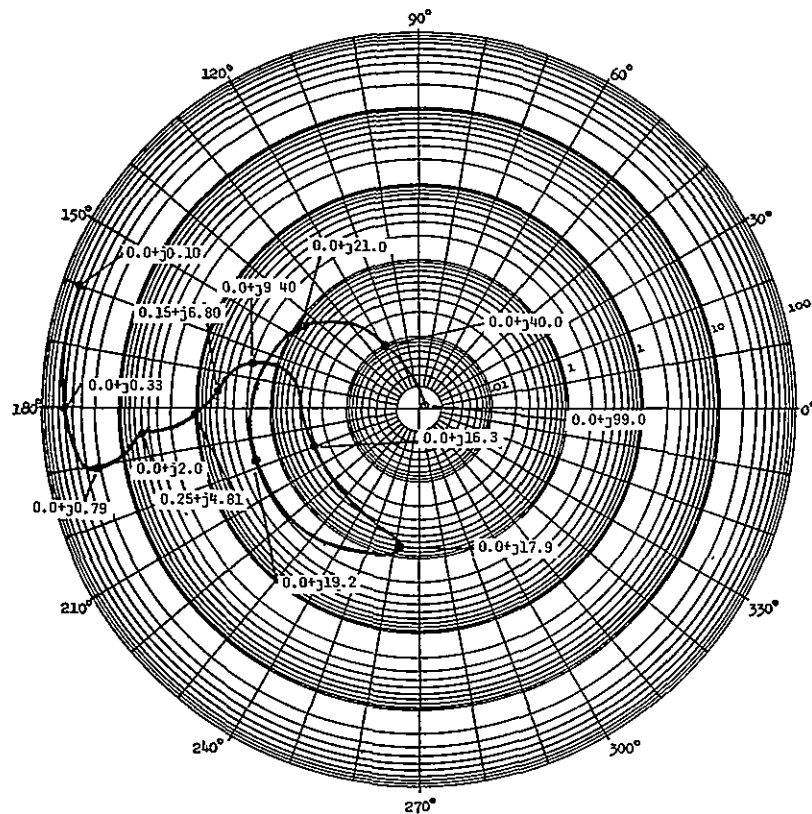


Figure 11. Frequency Response Representing Each Subsystem of the Uncompensated System of the Finned Vehicle Example.

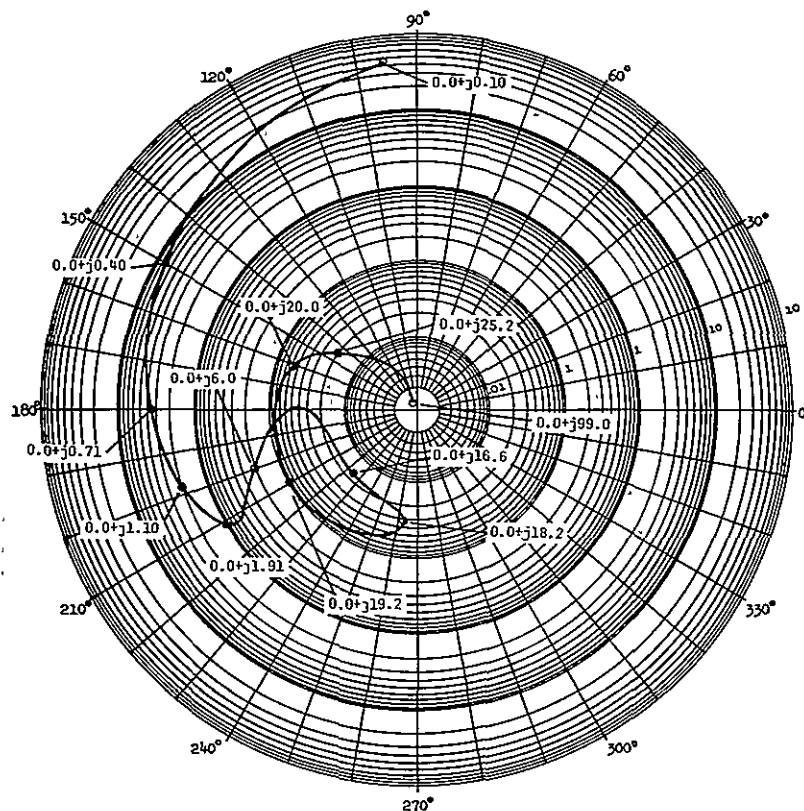


Figure 12. The Improved Compensated Frequency Response of Example 1.

Table 3. System Specifications for the Finned Vehicle Examples.

Margin Number	Margin Value	Complex RE[s]	Frequency IM[s]	Desired Margin	Margin Type	Active List
A. The Uncoupled System:						
Iteration No. 0		Subsystem 1, 2.				
1	42.2900	0.000	0.347	0.50	GM	No
2	0.5036	0.200	6.400	0.50	GM	Yes
3	7.5340	0.250	4.437	30.00	PM	Yes
4	0.2241	0.000	19.190	0.10	AM	Yes
Iteration No. 31						
1	3.1130	0.000	0.673	0.50	GM	No
2	30.0300	0.000	2.086	30.00	PM	No
3	0.0920	0.000	19.190	0.10	AM	No
B. The Coupled System:						
Iteration No. 0		Subsystem 1.				
1	42.2900	0.000	0.347	0.50	GM	No
2	0.6114	0.000	8.000	0.50	GM	Yes
3	11.0500	0.250	4.613	30.00	PM	Yes
4	0.2976	0.000	19.190	0.10	AM	Yes
		Subsystem 2.				
1	29.3000	0.000	0.347	0.50	GM	No
2	0.2010	0.200	3.600	30.00	PM	Yes
3	0.2360	0.000	19.190	0.10	AM	Yes
Iteration No. 50		Subsystem 1.				
1	2.4590	0.000	0.700	0.50	GM	No
2	28.0300	0.000	1.882	30.00	PM	No
3	0.0870	0.000	19.190	0.10	AM	No
		Subsystem 2.				
1	30.0880	0.000	0.606	0.50	GM	No
2	29.4700	0.000	1.882	30.00	PM	No
3	0.0960	0.000	19.190	0.10	AM	No

specified design objectives versus the final system specifications. Similar results were obtained with the Compensator Improvement Program by Mitchell and McDaniel in reference [11] for the single input, output system.

A Coupled Dual Input, Dual Output System

This example utilizes the same system as the previous illustration but in this case the plant subsystems are coupled with non-zero off-diagonal terms. Thus, the same uncompensated frequency response data is used to describe the diagonal terms of the plant matrix $[P(s)]$. Table 4 gives the frequency response data describing the coupling terms, that is, $P_{12}(s)$ and $P_{21}(s)$. The compensation matrix remains the same as described in equation (4.3); however, for generality, the compensator gain of the $G_{22}(s)$ element has been altered to a factor of 0.7 instead of unity.

Requiring the same design objectives as stated in equations (4.2), the improved compensation elements,

$$G_{11}(s) = \frac{(1.0 + 1.42713s)(5.0 + 6.52626s)}{(1.0 + 19.2239s)(5.0 + 1.25501s)} \quad (4.4)$$

and

$$G_{22}(s) = \frac{0.7(1.0 + 2.35364s)(5.0 + 4.07685s)}{(1.0 + 15.2297s)(5.0 + 1.01990s)}, \quad (4.5)$$

are obtained after 50 iterations. Entry B in Table 3 shows the desired objectives as compared to the final system specifications after the compensation improvement.

Table 4. Frequency Response Data Describing the Coupling Elements of the Plant Matrix $[P(s)]$ in the Coupled Finned Vehicle Example

Data Point	Complex Frequency		Uncompensated Plant Response			
	RE[s]	IM[s]	RE $[P_{12}]$	IM $[P_{12}]$	RE $[P_{21}]$	IM $[P_{21}]$
1	0.000	0.100	0.1000	0.0005	0.2001	0.0066
2	0.000	0.132	0.1000	0.0066	0.2002	0.0088
3	0.000	0.178	0.1000	0.0009	0.2003	0.0118
4	0.000	0.240	0.1001	0.0012	0.2006	0.0159
5	0.000	0.347	0.1001	0.0017	0.2013	0.0230
6	0.000	0.501	0.1003	0.0029	0.2028	0.0331
7	0.000	0.646	0.1004	0.0032	0.2046	0.0425
8	0.000	1.230	0.1014	0.0058	0.2161	0.0786
9	0.000	2.000	0.1035	0.0086	0.2400	0.1200
10	0.100	2.800	0.1062	0.0134	0.2750	0.1492
11	0.200	3.600	0.1087	0.0112	0.3105	0.1681
12	0.250	4.000	0.1099	0.0115	0.3276	0.1744
13	0.250	4.921	0.1123	0.0119	0.3630	0.1866
14	0.250	6.000	0.1147	0.0118	0.4002	0.1988
15	0.200	6.400	0.1154	0.0118	0.1259	0.1935
16	0.100	7.200	0.1168	0.0115	0.4356	0.1940
17	0.000	8.000	0.1176	0.0102	0.4520	0.1742
18	0.000	10.071	0.1201	0.0099	0.4952	0.1759
19	0.000	12.400	0.1215	0.0087	0.5241	0.1568
20	0.000	15.600	0.1227	0.0073	0.5485	0.1340
21	0.000	18.327	0.1233	0.0063	0.5613	0.1183
22	0.000	19.191	0.1234	0.0061	0.5644	0.1139
23	0.000	19.638	0.1235	0.0059	0.5658	0.1118
24	0.000	20.563	0.1236	0.0057	0.5686	0.1076
25	0.000	30.415	0.1243	0.0040	0.5850	0.0759
26	0.000	40.095	0.1246	0.0030	0.5912	0.0585
27	0.000	60.686	0.1248	0.0020	0.5061	0.0391
28	0.000	100.000	0.1249	0.0012	0.5986	0.0239

A Dual Input, Four Output System with
Coupled Characteristics

This example is representative of the Yaw/Roll Ascent Flight Control System for the Space Shuttle. The system is similar to that of Figure 2 with a plant possessing two control inputs and four outputs. The 23 discrete frequency response data chosen to describe the plant dynamics are listed in Table 5. The compensation matrix $[G(s)]$ given in Table 6 actually is designed for use on the space shuttle and exemplifies the complexity required in achieving a set of design objectives.

Given the design requirements of Table 7, the CIP produced the improved compensation matrix of Table 6 in 5 iterations, that is, 20 seconds of CPU time on the Univac 1108.

In summary, the CIP is a fast and effective design tool in the area of compensation improvement. For the Finned Vehicle examples, approximately 28K words of core storage is required; the Shuttle example executes in 32K of storage. In each example the programming time and computer time is minimal.

Table 6. Compensation Matrix $[G(s)]$ in Cascaded Factor Form for the Space Shuttle Example.

COMPENSATOR (1,1): GAIN = 1.0000					
COMPENSATOR COEFFICIENTS:					
ZA = .100000-01	ZB = .744312				
ZC = 1.00000	ZD = 1.02373	ZE = 8.16526			
ZC = 1.00000	ZD = 2.30558	ZE = 2.01630			
ZC = 1.00000	ZD = 1.22262	ZE = 4.44622			
ZC = 1.00000	ZD = 1.02373	ZE = 8.16526			
PA = .133000-01	PB = 1.18973				
PC = 1.00000	PD = 4.36098	PE = 18.9002			
PC = 1.00000	PD = 4.36098	PE = 18.9002			
PC = 1.00000	PD = 2.38040	PE = 3.99765			
PC = 1.00000	PD = 4.36098	PE = 18.9002			
COMPENSATOR (1,2): GAIN = .74500					
COMPENSATOR COEFFICIENTS					
ZA = 1.00000	ZB = -.996201				
ZA = .100000-01	ZB = 1.01163				
ZC = 1.00000	ZD = .582803	ZE = 8.16205			
ZC = 1.00000	ZD = .656982	ZE = 10.4091			
ZC = 1.00000	ZD = .576605	ZE = 2.04025			
ZC = 1.00000	ZD = 1.20585	ZE = 4.44328			
PA = 1.00000	PB = 50.0069				
PA = .133300-01	PB = .983323				
PC = 1.00000	PD = 5.20890	PE = 18.9033			
PC = 1.00000	PD = 5.99161	PE = 25.0002			
PC = 1.00000	PD = 5.32565	PE = 11.1114			
PC = 1.00000	PD = 2.39359	PE = 4.00060			
COMPENSATOR (1,3): GAIN = .74500					
COMPENSATOR COEFFICIENTS					
ZA = .000000	ZB = 51.0048				
ZA = .100000-01	ZB = .869534				
ZC = 1.00000	ZD = .583262	ZE = 8.16477			
ZC = 1.00000	ZD = .657081	ZE = 10.4118			
ZC = 1.00000	ZD = .582806	ZE = 2.04267			
ZC = 1.00000	ZD = 1.21041	ZE = 4.44575			
PA = 1.00000	PB = 50.0095				
PA = .133000-01	PB = 1.10453				
PC = 1.00000	PD = 5.21469	PE = 18.9011			
PC = 1.00000	PD = 5.99959	PE = 24.9981			
PC = 1.00000	PD = 5.33026	PE = 11.1092			
PC = 1.00000	PD = 2.39180	PE = 3.99822			
COMPENSATOR (2,4): GAIN = 1.0000					
COMPENSATOR COEFFICIENTS					
ZA = .300000-01	ZB = .995009				
ZC = 1.00000	ZD = .827221	ZE = 8.16655			
ZC = 1.00000	ZD = .992354	ZE = 6.25852			
ZC = 1.00000	ZD = 2.00871	ZE = 1.56803			
ZC = 1.00000	ZD = 1.20369	ZE = 4.45146			
PA = .400000-01	PB = 1.00117				
PC = 1.00000	PD = 4.34837	PE = 18.9029			
PC = 1.00000	PD = 4.34837	PE = 18.9029			
PC = 1.00000	PD = 3.33705	PE = 11.1106			
PC = 1.00000	PD = 2.39557	PE = 3.99576			

Table 6. (Continued) Compensation Matrix $[G(s)]$ in Cascaded Factor Form for the Space Shuttle.

COMPENSATOR (1,1): GAIN = 1.0000					
COMPENSATOR COEFFICIENTS					
ZA = .100000-01	ZB = 1.00000				
ZC = 1.00000	ZD = 1.00000	ZE = 8.16300			
ZC = 1.00000	ZD = 2.28600	ZE = 2.01400			
ZC = 1.00000	ZD = 1.20000	ZE = 4.44400			
ZC = 1.00000	ZD = 1.00000	ZE = 8.16300			
PA = .133000-01	PB = 1.00000				
PC = 1.00000	PD = 4.34800	PE = 18.9030			
PC = 1.00000	PD = 2.34800	PE = 18.9030			
PC = 1.00000	PD = 2.40000	PE = 4.00000			
PC = 1.00000	PD = 4.34800	PE = 18.9030			
COMPENSATOR (1,2): GAIN = .74500					
COMPENSATOR COEFFICIENTS:					
ZA = 1.00000	ZB = -1.00000				
ZA = .100000-01	ZB = 1.00000				
ZC = 1.00000	ZD = .571400	ZE = 8.16300			
ZC = 1.00000	ZD = .645200	ZE = 10.4100			
ZC = 1.00000	ZD = .571400	ZE = 2.04100			
ZC = 1.00000	ZD = 1.20000	ZE = 4.44400			
PA = .133000-01	PB = 50.0074				
PA = 1.00000	PB = 1.00000				
PC = 1.00000	PD = 5.21700	PE = 18.9030			
PC = 1.00000	PD = 6.00000	PE = 25.0000			
PC = 1.00000	PD = 5.33300	PE = 11.1111			
PC = 1.00000	PD = 2.40000	PE = 4.00000			
COMPENSATOR (1,3): GAIN = .74500					
COMPENSATOR COEFFICIENTS					
ZA = .000000	ZB = 51.0074				
ZA = .100000-01	ZB = 1.00000				
ZC = 1.00000	ZD = .571400	ZE = 8.16300			
ZC = 1.00000	ZD = .645200	ZE = 10.4100			
ZC = 1.00000	ZD = .571400	ZE = 2.04100			
ZC = 1.00000	ZD = 1.20000	ZE = 4.44400			
PA = 1.00000	PB = 5010074				
PA = .133000-01	PB = 1.00000				
PC = 1.00000	PD = 5.21700	PE = 18.9030			
PC = 1.00000	PD = 6.00000	PE = 25.0000			
PC = 1.00000	PD = 5.33300	PE = 11.1110			
PC = 1.00000	PD = 2.40000	PE = 4.00000			
COMPENSATOR (2,4): GAIN = .74500					
COMPENSATOR COEFFICIENTS					
ZA = .300000-01	ZB = 1.00000				
ZC = 1.00000	ZD = .857000	ZE = 8.16300			
ZC = 1.00000	ZD = 1.00000	ZE = 6.25000			
ZC = 1.00000	ZD = 2.00000	ZE = 1.56300			
ZC = 1.00000	ZD = 1.20000	ZE = 4.44400			
PA = .400000-01	PB = 1.00000				
PC = 1.00000	PD = 4.34800	PE = 18.9030			
PC = 1.00000	PD = 4.34800	PE = 18.9030			
PC = 1.00000	PD = 3.33300	PE = 11.1110			
PC = 1.00000	PD = 2.40000	PE = 4.00000			
COMPENSATORS HAVING ZERO CONTRIBUTION: (1,4); (2,1); (2,2); (2,3)					

Table 7. System Specifications for the Space Shuttle Example.

Margin Number	Margin Value	Complex RE[s]	Frequency IM[s]	Desired Margin	Margin Type	Active List
A. Subsystem No. 1, Iteration No. 0						
1	0.4163	0.000	0.0896	0.60	GM	Yes
2	53.7400	0.000	0.0240	30.00	PM	No
3	127.5000	0.000	0.0603	30.00	PM	No
4	24.2100	0.000	0.0679	30.00	PM	Yes
5	0.0074	0.000	0.3412	0.10	AM	No
6	0.0289	0.000	0.3469	0.10	AM	No
7	0.0097	0.000	0.6600	0.10	AM	No
8						
Subsystem No. 2, Iteration No. 0						
1	0.5941	0.000	0.0896	0.60	GM	Yes
2	36.5000	0.000	0.0321	30.00	PM	No
3	0.0885	0.000	0.3434	0.10	AM	Yes
4	0.0076	0.000	0.6600	0.10	AM	No
B. The Improved System Specifications						
Subsystem No. 1, Iteration No. 5						
1	0.6120	0.000	0.0927	0.60	GM	No
2	38.1800	0.000	0.0222	30.00	PM	No
3	143.9000	0.000	0.0616	30.00	PM	No
4	32.0100	0.000	0.0674	30.00	PM	No
5	0.0307	0.000	0.3469	0.10	AM	No
6	0.0098	0.000	0.4917	0.10	AM	No
7	0.0041	0.000	0.6600	0.10	AM	No
Subsystem No. 2, Iteration No. 5						
1	0.6075	0.000	0.0896	0.60	GM	No
2	31.3600	0.000	0.0339	30.00	PM	No
3	0.0845	0.000	0.3434	0.10	AM	No
4	0.0076	0.000	0.6600	0.10	AM	No

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

Conclusions

Because of the complexity of technology and control laws, the design of modern control systems has become increasingly complicated. In this exposition, the theory and associated numerical technique for achieving a computer-aided compensation design improvement algorithm for the multivariable control system have been presented. The technique developed is applicable to linear, time-invariant systems possessing multiple input, multiple output status whose plant characteristics are described by discrete open-loop frequency response data. The compensation matrix is entered as transfer functions of cascaded first and second order polynomials. The method was designed using a strict constraint algorithm to alleviate the inherent problems generally associated with soft constraint cost functionals. The objective of the Compensator Improvement Program is to modify in an iterative manner the free parameters of the compensation yielding a system that satisfies specified frequency response properties. The computer coding in the Fortran IV language has been included and the practical utility of the program illustrated with space related examples.

Chapter I contains a literature survey of the previous research on the automatic design problem based on theory developed

by classical design methods. Each of the aforementioned techniques has been ascertained successful for the author's specified control area, generally restricted to a single control input system; however, the Compensator Improvement Algorithm [7] appeared most readily applicable and available for extension to the multivariable control case with the objective of obtaining a suboptimal solution to the specified constraints of the control design.

The theoretical concepts of the design algorithm are developed in Chapter II. The mathematical derivations associated with the algorithm in determining the necessary closed-loop frequency response, gradient vectors, and hence, the directional vectors, have been deduced by exact means.

In Chapter III the objective of the schematic algorithm and the computational flow diagram were introduced. This chapter contains an explanation of the flow diagram referring to the theory and synopsis of the subprograms in Appendix A.

Pragmatic examples illustrate the effectiveness of the algorithm in Chapter IV. Three space related examples were presented: (1) a dual input, dual output system exemplifying no coupling; (2) a dual input, dual output system with coupling; (3) a Space Shuttle example with dual control inputs and four outputs. The results herein verify the utility of the Compensator Improvement Program as a practical method of compensation improvement.

In conclusion, this exposition has demonstrated that the classical control theory is amenable to systems with multivariable characteristics. An important benefit derived from the use of the frequency domain for linear time-invariant systems is the intuition it provides in determining the soundness of a system. The digital computer has made possible the application of classical techniques to the optimal design problem. The ultimate contribution of this research effort is the development and implementation of an algorithm to enhance compensator design of systems possessing multivariable control characteristics.

Recommendations

The employment of any digital computer algorithm as an aid in the aggregate design process is perhaps as much an art as the design process itself. The use of the computer-aided design program may free the engineer from many burdensome and time-consuming calculations, but it is the engineer who in essence must provide the framework in which to enter the compensation in order to achieve the desired control law. This then is perhaps the greatest limitation of any computer-aided control design; that is, there is no supplanting the awareness and judgement of an experienced control engineer.

More realistically, however, the Compensator Improvement Algorithm does possess minor limitations which could be reconciled. In particular,

1. CIP should be given the option of accepting either
discrete frequency response data or transfer function

information. With the transfer function option, CIP would calculate its own frequency response data.

2. Modifications should be made for CIP to accept prefilters; these filters would be user designated and not altered by the program.
3. CIP should be given the option of accepting sampled-data systems without the necessity of the engineer converting compensation into the W-plane; possibly CIP could be further extended to accept multirate sampling problems.
4. The practical utility of CIP could be extended by rendering the algorithm capable of producing a two phase optimization program: in particular, the present version of the algorithm would yield a design to meet a set of design objectives producing a feasible solution while continuing to satisfy the desired specifications. For the second phase, perhaps a gradient projection technique could be utilized in optimizing the necessary cost function.

In essence the major limitation of the CIP algorithm is its restriction to linear, time-invariant control systems. The aspects of extending the work to nonlinear systems have not been contemplated; this omission is regrettable since the occurrence of system uncertainty is always a possibility. In regard to the restriction

of time-invariance, the present CIP techniques are not amenable to time-varying systems.

Mississippi State University

Mississippi State, Mississippi 39762

July 25, 1977

APPENDICES

APPENDIX A

SUBPROGRAM SUMMARIES OF THE COMPENSATOR IMPROVEMENT PROGRAM

Introduction

It is the objective of this Appendix to provide the basic concepts in theory and/or programming techniques incorporated within each subroutine. With the enclosed information any efforts made in adaptations or modifications for solving related problems should be reduced significantly. The subprograms are presented in alphabetical order for easy reference.

Subroutine ADDPTS

In an effort to minimize the input data storage required and the corresponding computer time used in manipulating extensive data, the subprogram ADDPTS[9] generates additional frequency data. In particular, if the spacing of the original response data in the neighborhood of a critical point in the relative stability region becomes too large, this subprogram interpolates the given data in this neighborhood yielding a more accurate stability margin. This design philosophy is based on the continuous nature of the frequency response over the complete range of frequencies. The added frequency points are obtained in accordance with the routine INTER, an interpolation algorithm; a log type of interpolation is used in determining all magnitudes; whereas, phases are calculated by linear interpolation.

This subroutine also requires the routines CRT and EVAL for updating the frequency response at the data points. The routine DELETE is used in conjunction with ADDPTS to retain only the original data at each new iteration.

The following variables are designated for this subprogram:

Subprogram VariablesInput Variables:

- KPOINT - An integer variable used to denote the current number of data points.
- KIN - An integer variable that denotes the number of inputs to the controller.

- KOUT - Integer variable denoting the number of controller outputs.
- NB - An integer used as a counter representing the starting number of the margins to be investigated.
- NM(I) - An integer array representing the number of margins investigated for each subsystem.
- STBM(I) - A one dimensional real array containing the values of the stability margins.
- KPTS(I,J) - A two dimensional integer array containing the frequency numbers of the margins in accordance with a particular subsystem.
- CT(I,J) - A two dimensional complex array containing the overall frequency response of the system.
- G(I,J,K) - A three dimensional complex array containing the original discrete data frequency response of the plant system.
- GC(I,J,K) - A complex three dimensional array storing the compensator response evaluated at the specified frequency points.
- T(I,J,K) - A complex three dimensional array in which the transfer response $[GC] \cdot [G]$ is stored.
- OMEGA(I) - A one dimensional complex array containing the discrete frequency points.
- KPTMAX - An integer denoting the maximum number of discrete frequency points allowable on a single iteration.

- KGOBAK - An integer variable used to denote whether frequency points were added.
- NIT(I) - An integer array denoting the number of inactive margins for each subsystem.
- KINACT(I,J) - A two dimensional array containing the integer numbers corresponding to the frequencies of the inactive or satisfied margins for each subsystem.
- NML(I) - An integer array denoting the number of active margins detected per subsystem.
- KACT(I,J) - A two dimensional integer array denoting the frequency data numbers of the active or unsatisfied margins for each subsystem.
- KPOLD - An integer denoting the number of data points on the last iteration.
- KOLD(I) - A one dimensional integer array containing the previous data points.
- KSYM - An integer variable used to reference the frequency response of the particular subsystem being manipulated.

The following transient variables are used in the auxiliary subroutines CRT and EVAL and are not affected directly by this subprogram; for more information regarding these variables refer to the respective subroutine synopsis.

CRT: C1,CI,WORKI

EVAL: ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,
N1,N2,M1,M2,GAIN,KONT,A,B,C,D,E

Output Variables:

The following output variables are defined as their respective input counterparts, but as outputs have been updated to include the newly generated data points: $G(I,J,K)$, $GC(I,J,K)$, $T(I,J,K)$, $KPOINT$, $OMEGA(I)$, $KINACT(I,J)$, $KACT(I,J)$, $KOLD(I)$.

Subroutine CHANGE

The subprogram CHANGE is designed to change the individual compensator coefficients in an orderly manner to induce an improved system. The change in the compensator coefficients is made in accordance with the directional vector of the particular iteration.

Note that the compensator elements are described by the transfer functions in the form of cascaded first and second order factors, that is,

$$G(s) = (\text{GAIN}) \frac{\prod_{i=1}^{N1} (ZA_i + ZB_i s) \prod_{j=1}^{N2} (ZC_j + ZD_j s + ZE_j s^2)}{\prod_{i=1}^{M1} (PA_i + PB_i s) \prod_{j=1}^{M2} (PC_j + PD_j s + PE_j s^2)} \quad (A-1)$$

Recall that the ultimate goal of the Compensator Improvement Program is the design of compensation so that the measurements of the system performance are equal to or better than the system specifications. The design of the compensators can be expressed mathematically as the strict constraint problem:

$$\begin{aligned} & \text{Determine } \underline{x}^T \\ & \text{subject to } g_i(\underline{x}^T) \geq b_i, \quad i = 1, \dots, m \end{aligned} \quad (A-2)$$

where $\underline{x}$ is a vector of the n compensator coefficients. The functions $g_i(\underline{x}^T)$ contain measurements of the system performance in terms of the compensator coefficients; thus, these functions represent the stability and attenuation margins, as well as any constraints on the

compensator coefficients. The constants b_i are the system specifications.

Assuming that for equation (A-2) the trial solution vector at the k th iteration is $\underline{x}_k^T$, then a trial solution vector of a possible improved solution at the $(k + 1)$ th iteration is

$$\underline{x}_{k+1}^T = \underline{x}_k^T + h[\nabla G] \underline{a} \quad (A-3)$$

where $[\nabla G]$ is the Jacobian matrix evaluated at $\underline{x}_k$ and consists of all the active constraints, i.e., the functions $g_i(x_k) < b_i$. The scalar h is a normalized step size. The vector $\underline{a}$ is calculated as

$$\underline{a} = [\nabla G^T \nabla G]^{-1} \underline{c} \quad (A-4)$$

where $\underline{c}$ is a vector of weighting constants initially set at unity.

The routine DIRVEC determines the directional vector $\underline{d}$, where

$$\underline{d} = [\nabla G] \underline{a} \quad (A-5)$$

Thus the subprogram CHANGE ultimately applies the elements of this directional vector to its corresponding compensator coefficients weighted by the step size h .

The program variables are defined as follows:

Subprogram Variables

Input Variables:

ZA(I,J,K) - A real three dimensional array representing the constant terms of the first order factors of the compensation polynomials.

- ZB(I,J,K) - A real three dimensional array containing the first order factor coefficients of s .
- ZC(I,J,K) - A real three dimensional array representing the constant terms of the second order factors.
- ZD(I,J,K) - A three dimensional real array containing the s coefficients of the second order factors.
- ZE(I,J,K) - A real three dimensional array containing the s^2 coefficients of the second order factors.
- N1(I,J) - An integer array that denotes the number of cascaded first order factors.
- N2(I,J) - An integer array that denotes the number of cascaded second order factors.
- DV(I) - A real one dimensional array representing the directional vector $\underline{d}$ in equation (A-5).
- DEL - A real variable that denotes the step size h in (A-3).
- KKK - An integer used to count the number of elements in the directional vector.
- KIN - An integer denoting the number of controller inputs or sensed states.
- KOUT - An integer denoting the number of controller outputs.
- A,B,C - Parameter variables used to dimension the arrays by the number of maximum allowable cascaded factors.

Output Variables:

The following output variables are defined in the same manner as their respective input variables, but have been updated incrementally in accordance with the directional vector and step size: ZA(I,J,K), ZB(I,J,K), ZC(I,J,K), ZD(I,J,K), ZE(I,J,K).

Subroutine CRT

The subprogram CRT has two major functions as denoted by the input variable KEY. If KEY has been set to unity, the program calculates the closed-loop frequency response $[C(s)]$ in terms of the input vector, that is,

$$[C(s)] = [G(s)][P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] \quad . \quad (A-6)$$

Thus the open-loop frequency response of the k th system can be obtained by setting the k th diagonal element of $[H(s)]$ to zero, and by setting all the elements of $[R(s)]$ to zero except the k th element which is set to unity; the result is the frequency response between the k th input and the outputs when the k th loop is open.

If KEY is entered as zero, the program is designed to aid the subprogram PARTIAL in determining the partial term $\partial C_k(s)/\partial G_{ij}$. Actually, this term can be evaluated by the equation

$$\begin{aligned} \frac{\partial [C(s)]}{\partial G_{ij}} = & -[G(s)][P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1} [H(s)] \frac{[G(s)]}{G_{ij}} \cdot \\ & [P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] \\ & + \frac{\partial [G(s)]}{\partial G_{ij}} [P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] \quad (A-7) \end{aligned}$$

which bears semblance to equation (A-6). Then, $\partial C_k(s)/\partial G_{ij}$ is the k th element of (A-7) with the conditions aforementioned.

Other routines used by this subprogram are the complex matrix inversion and multiplication programs, MATINC and MATMUL

respectively. The variables are defined in the following list:

CRT Subprogram Variables

Input Variables:

- KEY - A switch variable which designates the subprogram mode: if KEY is set to unity, the response of (A-6) is returned; if KEY is set to zero, the partial term of (A-7) is calculated and returned.
- K - An integer variable denoting specific frequency point under consideration.
- KSTM - An integer variable representing the system under consideration, that is, k in the previous discussion.
- T(I,J) - A two dimensional complex array of the transfer response $[G(s)] \cdot [P(s)]$.
- KIN - An integer denoting the number of control inputs.
- KOUT - An integer denoting the number of controller outputs.
- P(I,J) - A two dimensional complex array of the plant response.
- I1 - An integer denoting the control input index of G_{ij} in (A-7).
- J1 - An integer denoting the control output index of the term G_{ij} in equation (A-7).
- Cl(I,J) - A two dimensional internal complex array containing the total frequency response at a particular frequency point.
- CI(I,J) - An internal two dimensional array storing the inverse of the complex frequency response Cl(I,J).

WORKI(I,J) - A two dimensional complex array for internal subroutine use.

A,B,D - Variables denoting dimension allocations.

Output Variables:

CT(I) - A one dimensional complex array representing the response $[C(s)]$ in equation (A-6).

PCG - A complex variable representing the partial term $\partial C_k(s)/\partial G_{ij}$ in the previous discussion.

Subroutine DELETE

The subprogram DELETE is designed in conjunction with the ADDPTS routine in an effort to save computer storage and time in manipulating unnecessary data. In particular, this subprogram deletes the extra frequency points and their corresponding response terms generated by the ADDPTS routine when the maximum number of allowable points has been generated; for accuracy, the original input data is always retained.

The routine uses the following variables:

Subprogram Variables

Input Variables:

- KPOINT - An integer counter to denote the number of frequency points.
- KIN - An integer denoting the number of controller inputs or the measured states.
- KOUT - An integer variable representing the number of controller outputs or plant inputs from the controller.
- OMEGA(I) - A one dimensional complex array of frequency terms.
- G(I,J,K) - A three dimensional complex array consisting of the frequency response describing the plant system.
- NIT(I) - An integer array denoting the number of inactive or satisfied margins detected.
- KINACT(I,J) - A two dimensional integer array of the frequency numbers of the corresponding inactive margins for each subsystem.

- NML(I) - An integer array denoting the total number of active
 or unsatisfied margins detected per subsystem.
- KACT(I,J) - A two dimensional integer array of the frequency
 data of the active margins.
- ITER - An integer representing the present iteration.
- KPOLD - An integer denoting the total number of frequency
 points.
- KOLD(I) - An integer array of the original frequency points
 reference numbers.
- KNEW(I) - An integer array of the generated frequency numbers.
- A,B,C,D,E - Dimension allocations; set by parameter statement
 in the main program.

Output Variables:

The following output variables correspond to their respective input variables, but have been updated by the subprogram deleting the extra data: OMEGA(I), G(I,J,K), KINACT(I,J), KACT(I,J), KOLD(I).

Subroutine DIRVEC

The objective of the subprogram DIRVEC is to calculate the directional vector of the constraint improvement algorithm. The directional vector $\underline{d}$ is calculated as

$$\underline{d} = [\nabla G] \underline{a} \quad (\text{A-8})$$

where ∇G is a matrix of order (n,m) whose columns consist of the gradients of the active constraints. The m component column vector $\underline{a}$ is determined by

$$\underline{a} = [\nabla G^T \nabla G]^{-1} \underline{c} \quad (\text{A-9})$$

where $\underline{c}$ is a m component column vector whose elements are all positive. The order m corresponds to the number of compensator coefficients. Auxiliary subprograms include the matrix inversion for real variables MATINV and the matrix multiplication MATMUL routines.

Definitions of the input and output variables are as follows:

Subprogram Variables

Input Variables:

- | | |
|--------|--|
| G(I,J) | - A two dimensional array whose columns are comprised of the gradients of the active constraints. |
| NM | - Integer value used to designate the number of columns in G, that is, the number of active constraints. |
| KPARC | - Integer variable that represents the number of rows in G, that is, the number of compensator coefficients. |

WEIGHT(I) ~ A real one dimensional array that contains the
 column matrix $\underline{c}$ of equation (A-9).

E,F,Z,H ~ Parameter variables for allocating dimension storage.

The transient variables, AI, AI, and WORKR, are used in the
auxiliary subprograms MATINV and MATMUL and are not affected direct-
ly in this routine.

Output Variables:

DV(I) ~ A real one dimensional array which corresponds to the
 directional vector $\underline{d}$ in (A-8).

Subroutine EVAL

The subprogram EVAL is designed to evaluate the compensator matrix $[G(s)]$ at each discrete frequency point. Recall the compensation elements are polynomials of the form of cascaded first and second order factors, that is,

$$G(s) = (\text{GAIN}) \frac{\prod_{i=1}^{N1} (ZA_i + ZB_i s) \prod_{j=1}^{N2} (ZC_j + ZD_j s + ZE_j s^2)}{\prod_{i=1}^{M1} (PA_i + PB_i s) \prod_{j=1}^{M2} (PC_j + PD_j s + PE_j s^2)} \quad (A-10)$$

The subprogram utilizes the polynomial evaluation routine POLEV.

This subroutine also calculates the transfer relation $[G(s)] \cdot [P(s)]$ for each frequency point using the matrix multiplication program MATMUL.

The subprogram uses the following variables:

Subprogram Variables

Input Variables:

- XF - A complex variable denoting the discrete frequency point.
- G(I,J,K) - A three dimensional complex array containing the original discrete data frequency response of the plant.
- K - An integer variable used to denote the current number of the frequency data point.
- KIN - An integer variable denoting the number of inputs to the controller.

- KOUT - An integer that denotes the number of controller outputs.
- ZA(I,J,K) - A real three dimensional array containing the constant terms of the first order factors of the numerator.
- ZB(I,J,K) - A real three dimensional array containing the first order factor coefficients of s in the numerator.
- ZC(I,J,K) - A real three dimensional array representing the constant terms of the second order numerator factors.
- ZD(I,J,K) - A real three dimensional array containing the s coefficients of the second order factors of the numerator.
- ZE(I,J,K) - A real three dimensional array storing the s^2 coefficients of the second order factors in the numerator.
- PA(I,J,K) - A real three dimensional array containing the constant terms of the first order denominator factors.
- PB(I,J,K) - A real three dimensional array containing the first order denominator factor coefficients of s .
- PC(I,J,K) - A real three dimensional array representing the constant terms of the second order denominator factors.
- PD(I,J,K) - A real three dimensional array containing the s coefficients of the second order factors in the denominator.
- PE(I,J,K) - A real three dimensional array storing the s^2 coefficients of the second order denominator factors.
- N1(I,J) - An integer array that denotes the number of cascaded first order factors in the numerator.
- N2(I,J) - An integer array that denotes the number of second order cascaded factors in the numerator.

- M1(I,J) - An integer array denoting the number of cascaded first order factors in the denominator.
- M2(I,J) - An integer array that denotes the number of second order cascaded factors in the denominator.
- GAIN(I,J) - A two dimensional real array containing the DC gain of each compensator polynomial.
- KONT(I,J) - A two dimensional integer array designating whether the DC gain for the particular channel is allowed to vary: if KONT is unity, the gain may vary; if KONT is two, the gain is not allowed to vary.
- A,B,C,D - These variables are defined by a parameter statement in the main program and designate maximum storage allocations in regards to the order of the arrays.

Output Variables:

- GC(I,J,K) - A three dimensional complex array storing the compensator response evaluated at specified frequency points.
- T(I,J,K) - A complex three dimensional array in which the transfer response $[GC] \cdot [G]$ is stored.

Subroutine GAINMG

The purpose of the subprogram GAINMG is to locate and calculate the gain margins of a system represented by a discrete open-loop frequency response. With f_i as the i th frequency specified, the corresponding complex frequency response can be represented in real and imaginary terms as GR_i and GI_i respectively. Thus the following sequence can be formed to detect the occurrence of a gain margin:

$$U_i = GI_i \cdot GI_{i-1} \quad (A-11)$$

A gain margin is located whenever U_i becomes either negative or zero. The frequency number of the gain margin is taken as i or $i-1$ depending on whether $|GI_i| > |GI_{i-1}|$ or $|GI_i| \leq |GI_{i-1}|$; the gain margin is calculated as

$$STBM = |1. + GR_k + jGI_k| \quad , \quad (A-12)$$

where k is either i or $i-1$.

The following variables are designated for the subprogram:

Subprogram Variables

Input Variables:

- OMEGA(I) - A complex one dimensional array that contains the specified frequencies in ascending order for describing the system.
- GTOTAL(I) - A complex one dimensional array containing the compensated open-loop frequency response for the i th frequency.

- KPOINT - An integer number of frequency points used to describe the open-loop frequency response of the system.
- FQMIN - The lowest frequency for which gain margins are detected.
- FQMAX - The largest frequency for which the gain margins are to be detected.
- NM - The integer used as a counter for the number of margins located. For example, assume NM is initially 2 and this program locates 3 margins; these margins would be labelled as margins 3, 4, and 5 respectively.

Output Variables:

- NM - This is the number that designates the last gain margin found.
- KPTS(I) - A one dimensional integer array that contains the frequency members of the margins found.
- STBM(I) - A one dimensional real array that contains the margin values corresponding to the frequency pointer KPTS. These margins are measured in terms of distances from the point $(-1 + j0)$ in the complex $GH(j\omega)$ plane.

Subroutine INTER

INTER is a subprogram designed in conjunction with the ADDPTS routine to interpolate specified input data, thereby generating new data. The algorithm is designed to yield a log type of interpolation in determining all magnitudes; whereas, phases are calculated by a linear interpolation scheme.

The variables are defined in the listing:

Subprogram VariablesInput Variables:

- S - A complex number consisting of the lower bound of the quantity to be interpolated.
- T - A complex number consisting of the upper quantity bound of the interpolation.

Output Variable:

- R - A complex number representing the resultant of the interpolation.

Subroutine MATINC

Subroutine MATINC determines the inverse of a matrix of complex elements by the Gauss-Jordan reduction method. It is assumed that no diagonal elements of the original matrix are zero. If in applying the reduction procedure the magnitude of the i th element of the i th pivot row is of magnitude less than 1.0×10^{-25} , the inverse matrix is assumed nonexistent.

The input, output variables are defined as:

Subprogram Variables

Input Variables:

- XX(I,J) - A complex two dimensional square array whose inverse is desired.
- N - An integer denoting the number of rows and columns in matrix XX(I,J).
- X(I,J) - A complex array containing the generated augmented matrix.
- A,B,C - Parameter variables denoting storage allocation.

Output Variables:

- YY(I,J) - A complex two dimensional array that contains the inverse of the XX(I,J) array.
- IER - The error code of the subprogram. If IER is zero, no error was incurred; if IER is unity, the matrix is assumed to be singular.

Subroutine MATINV

The subprogram MATINV uses the Gauss-Jordon reduction method in determining the inverse of a matrix of real elements. The procedure and program variables of this routine are defined in the same manner as in the subprogram MATINC but with application to the real matrix $XX(I,J)$ and its inverse $YY(I,J)$ composed of real elements only.

Subroutine MATMUL

Subroutine MATMUL is designed to determine the product of a matrix A of order (n, ℓ) by a matrix B of order (ℓ, m) . The elements c_{ij} of the resultant matrix C of order (n, m) are obtained by the equation

$$c_{ij} = \sum_{k=1}^{\ell} a_{ik} b_{kj} \quad . \quad (A-13)$$

This subprogram is designed to operate on either real or complex matrices as specified by the input variable NC. The input, output variables are designated as follows:

Subprogram Variables

Input Variables:

- AC(I,J) - A complex two dimensional array representing the matrix A as aforementioned when the subprogram is used in the complex mode.
- BC(I,J) - A complex two dimensional array representing the aforementioned matrix B.
- AR(I,J) - A real two dimensional array corresponding to matrix A in the previous discussion when the subprogram is used in real term mode only.
- BR(I,J) - A real two dimensional array corresponding to a real matrix B.
- N - An integer variable denoting the number of rows in the matrix A.
- L - An integer variable denoting the number of columns in A.

- LI - An integer variable denoting the number of rows in matrix B in the above discussion.
- M - An integer variable that denotes the number of columns in matrix B.
- ND - An integer variable used to designate the proper storage placement when multiplying three dimensional matrices; for the two dimensional case set ND to unity.
- NC - An integer which designates whether the program is to multiply real or complex matrices. If NC is zero, the complex matrices AC, BC, CC are used; if NC is unity, the real matrices AR, BR, CR are manipulated.
- A,B,C, - These variables are defined by a parameter statement in the main program and designate maximum storage capabilities in regard to the order of the matrices.

Output Variables:

- CC(I,J,ND) - A complex three dimensional array that contains the complex elements c_{ij} in equation (A-13).
- CR(I,J,ND) - A real three dimensional array containing the resultant matrix C when the subprogram is in real mode.

11-2

Subroutine NYQUIST

The subprogram NYQUIST determines the number of closed-loop poles of a closed-loop system inside a certain enclosed contour of the s-plane. The number of closed-loop poles within the contour is from the relation

$$Z = P + N, \quad (A-14)$$

where Z is the number of closed-loop poles, i.e., the number of roots of the characteristic equation inside the contour; p is the number of open-loop poles within the contour, and N is the algebraic sum of the encirclements around the $(-1 + j0)$ point by the frequency response. Note the encirclements around the $(-1 + j0)$ point are assumed positive if clockwise and negative if counterclockwise.

The NYQUIST encirclements are counted by application of the equation:

$$N = \text{INTEGER} \left\{ \pm \frac{1}{8} + \frac{P_c}{2} + \sum_{i=2}^M \{ \text{ANGLE}[1 + G(s_i)] - \text{ANGLE}[1 + G(s_{i-1})] \} / \pi \right\} \quad (A-15)$$

where the operator INTEGER yields only the integer portion of the calculation in brackets, and P is the number of poles on the contour. It is assumed that the contour is symmetric with respect to the real axis in the s-plane; hence the frequency response $G(s)$ is symmetric about the real axis of the $G(s)$ -plane. Thus in equation (A-15), only frequency points on that portion of the

contour in the upper half-plane are used in the evaluation. The fraction $\pm 1/8$ is used to account for the fact that the frequencies s_1 and s_M , the first and last frequency points respectively, may not actually be on the real axis. It is assumed that the points are chosen so that the sum of the angles is within $\pi/4$ radians of the correct value. The positive and negative signs are chosen in agreement with the the sign of the summation respectively.

The input, output variables are defined as follows:

Subprogram Variables

Input Variables:

- N - An integer variable that denotes the number of frequency response points supplied by the user, i.e., M in (A-15).
- G(I) - A complex one dimensional array containing the frequency response $G(s)$ where s is chosen along the contour.
- NRHP - An integer number of the open-loop poles inside the contour, i.e., P in the previous discussion.
- NCON - An integer number of open-loop poles located on the contour.
- FMAX - A real variable denoting the maximum frequency range.
- F(I) - A complex one dimensional array containing the specified frequency points.

Output Variables:

- NCIRL - An integer representing the number of encirclements around the $(-1 + j0)$ point.
- NZ - An integer of the number of poles of the closed-loop system inside the contour.

Subroutine OUTPT

The purpose of this subprogram is to output certain information at various stages of the main program. Three areas of information available for output are: Compensator Information, Frequency Response Information, and Stability Margin Data.

If the user desires data on the compensation matrix, the selector variable N is set to zero and the program outputs the compensator values at the last iteration.

By setting the selector variable to unity, the overall frequency response data is the output.

For investigating the stability margins, the selector variable is set to two. The corresponding output data includes:

1. The margin numbers
2. The frequency where each margin occurs
3. The value of each margin
4. The desired value of each margin
5. The type of margin, i.e., phase margin (P), gain margin (G), stability margin (S), or attenuation margin (A)
6. The directional vector at the last iteration.

Subroutine PARTAL

The Compensator Improvement Program is given the ability to determine the necessary partial derivatives of the various active margins, assuming a frequency response along a general contour, by the subprogram PARTAL. In order to develop this subroutine, the theoretical derivations of Section 3 of Chapter II, were implemented to yield these partial derivatives.

Recall that in designing compensation, CIP searches the compensated frequency response $C(j\omega)$ over specified ranges of frequency to determine which margins do not satisfy the desired values. For the unsatisfied or active margins, the design specifications are converted to distances between certain critical points of the open-loop frequency response and particular points in the corresponding complex plane; that is, the typical objective function for the k th open-loop system is

$$d = \{[A + C_k(j\omega)][A + C_k(j\omega)]^*\}^{\frac{1}{2}} \quad (A-16)$$

where $C_k(j\omega)$ is the k th system's compensated response and A is the point in the complex plane from which the specification is measured. Generally, point A is chosen as the $(-1.0 + j0.0)$ point for stability margins and $(0.0 + j0.0)$ for attenuation margins.

PARTAL determines the gradients of these design objectives with respect to the compensator coefficients of the controller. Thus, the partial derivatives of d with respect to some parameter w is

$$\frac{\partial d}{\partial w} = \operatorname{Re} \left\{ [A + C_k(j\omega)]^* \frac{\partial C_k(j\omega)}{\partial w} \right\} \div d \quad . \quad (\text{A-17})$$

Evaluation of (A-17) depends largely on determining $\partial C_k(j\omega)/\partial w$ accurately. Using the chain rule this becomes

$$\frac{\partial C_k(j\omega)}{\partial w} = \frac{\partial C_k(j\omega)}{\partial G_{ij}} \cdot \frac{\partial G_{ij}}{\partial w} \quad (\text{A-18})$$

where G_{ij} is the element of the controller $[G(s)]$ in which the free parameter w appears.

As derived in Chapter II, the first term in equation (A-18) may be evaluated by taking the k th element of

$$\begin{aligned} \frac{\partial [C(s)]}{\partial G_{ij}} &= -[G(s)][P(s)]\{I + [G(s)][P(s)][H(s)]\}^{-1}[H(s)] \frac{\partial [G(s)]}{\partial C_{ij}} \cdot \\ &\quad [P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] \\ &\quad + \frac{\partial [G(s)]}{\partial G_{ij}} [P(s)]\{I + [H(s)][G(s)][P(s)]\}^{-1}[R(s)] \end{aligned} \quad (\text{A-19})$$

where the k th diagonal element of $[H(s)]$ is set to zero and all the elements of $[R(s)]$ are set to zero except the k th which is set to unity. Note also that $\partial [G(s)]/\partial G_{ij}$ is a zero matrix except for the (ij) th element which is unity.

The second term in (A-18), $\partial G_{ij}/\partial w$, is derived in Chapter II. Assuming the (ij) th element of $[G(s)]$ is composed of a cascaded arrangement of transfer functions, i.e.,

$$G_{ij}(s) = \prod_{k=1}^K G_{ijk}(s) \quad (A-20)$$

where K is the number of cascaded elements. The ℓ th cascaded element of the (ij) th compensator of $[G(s)]$ has the general form

$$G_{ij}(s) = \frac{\sum_{m=0}^M x_{ij\ell m} s^m}{\sum_{n=0}^N y_{ij\ell n} s^n} \quad (A-21)$$

Then, the free parameters of this element are the x 's and y 's. If w in (A-17) is the p th numerator coefficient of (A-21), then

$$\frac{\partial G_{ij}(s)}{\partial x_{ij\ell p}} = G_{ij}(s) \frac{+s^p}{\left(\sum_{m=0}^M x_{ij\ell m} s^m \right)} \quad (A-22)$$

Similarly, if w represents the p th denominator coefficient of (A-21), then

$$\frac{\partial G_{ij}(s)}{\partial y_{ij\ell p}} = G_{ij}(s) \frac{-s^p}{\left(\sum_{n=0}^N y_{ij\ell n} s^n \right)} \quad (A-23)$$

Equations (A-16), (A-18), (A-22), and (A-23) indicate how the needed partial derivatives can be calculated. The subprogram PARTIAL implements these equations including the necessary logic for determining the perturbation points A and the orderly arrangement of the terms of the partials. The subprogram also performs the necessary

manipulations in calling the subroutine CRT which is used to determine $C_k(j\omega)$ in equation (A-16).

The input/output variables are defined in the following list. Note the compensator coefficients and related data enter the subprogram through a common block.

Subprogram Variables

Input Variables:

- OMEGA(I) - A complex one dimensional array that contains the specified frequencies in ascending order for describing the system.
- NFREQ - An integer variable denoting the number of active margins to be improved.
- CT(I) - A one dimensional complex array represented by the compensated closed-loop frequency response $C(s)$ in equation (A-19).
- KPTS(I) - An integer array used as a pointer to denote the frequency number of the margin investigated.
- TYPE(I) - A real array used to denote the type of margin being investigated.
- T(I,J,K) - A three dimensional complex array containing the transfer product of the compensation $G(s)$ and the plant response $P(s)$ for the specified frequencies.
- P(I,J,K) - A complex three dimensional array describing the open-loop frequency response of the plant.
- G(I,J,K) - A complex three dimensional array representing the compensation evaluated at the specified frequencies.

KSYM - An integer used for addressing the proper system
 arrays.
 PFX(I,J), - A two dimensional real array containing the partials
 PFY(I,J) of the numerator and denominator compensators respec-
 tively.
 A,B,C,D, - Parameter variables denoting dimension allocations.
 E,F,Z

The following variables are used to describe the compensation
 polynomials; please refer to the Subprogram EVAL for a complete
 description: KIN,KOUT,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N2,M1,M2,
 GAIN,KONT.

The transient variables apply to the following auxiliary sub-
 program and are not affected directly in this routine:

Subprogram CRT: C1,CI,WORKI

Output Variables:

NPARC - An integer variable that represents the number of
 partials determined.
 PG(I,J) - A two dimensional real array representing the $\partial d/\partial w$
 in (A-17).

Subroutine PHASEM

The subprogram PHASEM is used to detect and calculate the phase margins of an open-loop control system represented by a discrete frequency response. The open-loop frequency response is assumed to be given in terms of real and imaginary values. In particular given the i th frequency as f_i , the corresponding real and imaginary parts of the frequency response are GR_i and GI_i . The phase margins occur at the real zero crossings of the sequence:

$$S_i = 1.0 - |GR_i + jGI_i|^2 \quad . \quad (A-24)$$

Next the following sequence is formed:

$$U_i = S_i \cdot S_{i-1} \quad . \quad (A-25)$$

If $U_i \leq 0$, then either S_i or S_{i-1} is a zero or S_i has made a zero crossing. Regardless of which condition has occurred, the frequency number of the phase margin is chosen as i or $i-1$ depending on the smaller magnitude of S_i or S_{i-1} . The corresponding margin is calculated as

$$STBM = |1.0 + GR_k + jGI_k| \quad (A-26)$$

where k is either i or $i-1$ as mentioned above.

The following variables are defined for this program:

Subprogram Variables

Input Variables:

- OMEGA(I) - A complex one dimensional array that contains the specified frequencies in ascending order for describing the system.
- GTOTAL(I) - A complex one dimensional array containing the compensated open-loop frequency response for the Ith specified frequency point.
- KPOINT - The integer number of frequency points used to describe the open-loop frequency response of the system.
- FQMIN - The lowest frequency for which the particular margins are to be determined.
- NM - The integer used as a counter for the number of margins located.

Output Variables:

- NM - This is the number that designates the last margin found.
- STBM(I) - A one dimensional real array that contains the margin values corresponding to the frequency pointer KPTS(I). These margins are measured in terms of distances from the $(-1 + j0)$ point in the complex $GH(j\omega)$ plane.

Subroutine POLEV

The purpose of this subprogram is to evaluate polynomials at specified frequency points. The frequency data is complex in nature. The routine is designed with an internal subfunction to avoid inaccuracies in raising a complex variable to a power. The input, output variables are defined as follows:

Subprogram Variables

Input Variables:

- FW(I) - A one dimensional real array containing the polynomial coefficients in ascending order.
- K - An integer denoting the order of the polynomial.
- X - The complex variable of evaluation.

Output Variables:

- F - The complex resultant of the evaluation.

Subroutine SRMINS

The purpose of this subprogram is to calculate the maxima or minima of a discrete data open-loop frequency response with respect to a chosen point along the real axis. For example, assume GR_i and GI_i are the real and imaginary response terms corresponding to the i th frequency point. The following sequence is formed:

$$U_i = |P + GR_i + jGI_i|^2 \quad (A-27)$$

where P represents the negative point of investigation located on the real axis. Another sequence is generated as follows:

$$V_i = U_i - U_{i-1} \quad (A-28)$$

If $V_i \cdot V_{i-1} \leq 0$ and $V_{i-1} > 0$, then the $(i-1)$ frequency point corresponds to a relative maximum with respect to point P . Otherwise, if $V_i \cdot V_{i-1}$ is less than or equal to zero and $V_{i-1} < 0$, then the $(i-1)$ frequency is a relative minimum with respect to P .

The definitions of the variables are the same as those in the routine PHASEM with the following additional input variables:

Subprogram Variables

Input Variables:

- P - The negative value of the real axis point for which maxima or minima are to be located.
- N - An integer variable to determine whether the program is to determine maxima or minima. Maxima are investigated if N is unity; minima are found if N equals -1 .

Subroutine XCHECK

The XCHECK routine is not designed to check for damping ratio violations, but is necessary to assure that compensator poles and zeros on the specified zeta boundaries do not result in a zeta violation upon incrementation. The technique employed for designing the compensation is to adjust the directional vector by zeroing the corresponding partial vector terms until the directions of movement of the above mentioned poles are within the defined sector. In this effort, the subsystem values of the zeta damping factor and the undamped natural frequency ω_n must be calculated and compared to the user specifications.

Recall that the form of the compensation used by CIP is cascaded first and second order factors. In the case of first order factors if either of the coefficients is negative the program defaults with an error signal denoting the occurrence of a zeta violation, and, consequently, the run is terminated automatically; this can only occur on the first iteration because on succeeding iterations the subprogram, YCHECK, assures that a zeta violation cannot occur. The assumption here is that the user has erred in his initial selection of compensation.

If either of the coefficients is zero, the result is a root at the origin or at infinity. In this case the avoidance of a zeta violation is assured by forcing the corresponding terms of the directional vector to be nonnegative. This is implemented by first checking the signs of the corresponding terms of the directional

vector; if these signs are negative, the corresponding terms of the partial vector are zeroed and the directional vector is recomputed. On each iteration the number of variable coefficients is reduced by the number of terms in the directional vector forced to zero in this manner. If the number of variable coefficients becomes less than the number of active margins the CIA will fail and the run will terminate automatically.

In the case of second order factors the program calculates the ω_n and zeta as defined by the following second order factor:

$$T(s) = s^2 + 2\zeta\omega_n s + \omega_n^2 \quad . \quad (A-29)$$

This equation can be related to the CIP form of second order factors as:

$$T(s) = a_0 + a_1 s + a_2 s^2 \quad (A-30)$$

Comparing (A-29) and (A-30) it is easily seen that

$$\omega_n = \sqrt{a_0/a_2} \quad , \quad (A-31)$$

$$\zeta = (a_1)/(2\omega_n \cdot a_2) \quad . \quad (A-32)$$

The damping ratio ζ is checked to determine if a zeta violation has occurred. If such an occurrence is detected, as with first order factors the program is automatically terminated.

If the value of ζ indicates roots of the second order factor on the ζ boundaries, the next step is to determine whether these

roots will result in zeta violations upon incrementation of the compensator coefficients. This is accomplished by checking the sign of the change in ζ from incrementation.

The change in zeta with respect to the change in the coefficients of (A-30) is

$$\Delta\zeta = \frac{1}{2\omega_n} \left[-\frac{a_1}{2a_0 a_2} \Delta a_0 + \frac{1}{a_2} \Delta a_1 - \frac{a_1}{2a_2^2} \Delta a_2 \right] \quad (\text{A-33})$$

where Δa_0 , Δa_1 , and Δa_2 are the elements of the directional vector corresponding to a_1 , a_2 , and a_3 . If $\Delta\zeta$ is negative a zeta violation is inevitable. In order to avoid such an occurrence, associated terms of the partial vectors are zeroed and the directional vector is recomputed. This is continued until $\Delta\zeta$ is nonnegative. As with the first order factors the number of variable coefficients is reduced.

The program variables are defined as follows:

Subprogram Variables

Input Variables:

- KIN - An integer variable that denotes the number of inputs to the controller.
- KOUT - Integer variable denoting the number of controller outputs.
- ZA(I,J,K) - A real three dimensional array representing the constant terms of the first order factors.
- ZB(I,J,K) - A real three dimensional array containing the first order factor coefficients of s.

- ZC(I,J,K) - A real three dimensional array representing the constant terms of the second order factors.
- ZD(I,J,K) - A three dimensional real array containing the s coefficients of the second order factors.
- ZE(I,J,K) - A real three dimensional array containing the s^2 coefficients of the second order factors.
- N1(I,J) - An integer array that denotes the number of cascaded first order factors for each subsystem.
- N2(I,J) - An integer array that denotes the number of cascaded second order factors for each subsystem.
- DV(I) - A real one dimensional array representing the directional vector d in equation (A-5).
- ZETA - A real variable that denotes the desired minimum damping ratio.
- KKK - An integer used to count the number of elements in the directional vector.
- PG(I,J) - A two dimensional array containing the partial $\partial d / \partial w$ in equation (A-17).
- KRE - A program control integer.
- LPV - An integer denoting the number of variables allowable for change.
- NAM - An integer variable that represents the number of active, that is, unsatisfied margins.
- NRTR1 - An integer denoting whether first order factors are constrained to the left half GH(jw) plane: if 1, the factors are unconstrained; otherwise, constrained.

NRTR2 - Same as NRTR1 but applying to second order factors.
A,B,C,D,E - Parameter variables used to dimension the arrays by
 the number of maximum allowable elements.

Output Variables:

The following output variables are defined in the same manner
as their respective input variables, but have been updated in the
subprogram: PG(I,J), KKK, KRE, LPV.

Subroutine YCHECK

The subprogram YCHECK is designed to detect compensation poles or zeros that have been forced outside the allowable damping ratio sector due to incrementation of the coefficients. If such is the case, the routine selects a maximum step size which will inhibit the zeta boundary violation while continuing to produce an improved solution.

Recall that the CIP compensator data is described by transfer functions in cascaded first and second order factors. Similar to the XCHECK subprogram, YCHECK examines the first order compensator factors for possible right half plane roots. If such a root is found, the program reduces the step size until the root is marginally in the left half plane.

Assuming that all the first order roots are now in the left half plane, the subprogram proceeds to investigate the second order factors for possible zeta boundary violations. A typical second order factor of the form

$$T(s) = a_0 + a_1s + a_2s^2 \quad (A-34)$$

yields the relation for the undamped natural frequency

$$\omega_n = \sqrt{a_0/a_2} \quad , \quad (A-35)$$

and the zeta damping ratio as

$$\zeta = (a_1)/(2\omega_n \cdot a_2) \quad . \quad (A-36)$$

Upon comparison with the user specified damping ratio, if a zeta violation is incurred, the deincrementation of the second order coefficients continues iteratively until no violation occurs. From this increment a new step size is calculated and the routine returns to the main program to determine the compensator coefficients in accordance with this step size.

The input and output variables are defined in the following list.

Subprogram Variables

Input Variables:

- KIN ~ An integer variable that denotes the number of inputs to the controller.

- KOUT ~ Integer variable denoting the number of controller outputs.

- ZA(I,J,K) ~ A real three dimensional array representing the constant terms of the first order factors.

- ZB(I,J,K) ~ A real three dimensional array containing the first order factor coefficients of s .

- ZC(I,J,K) ~ A real three dimensional array representing the constant terms of the second order factors; i.e., a_0 in (A-34).

- ZD(I,J,K) ~ A real three dimensional array containing the s coefficients of the second order factors; i.e., a_1 in (A-34).

- ZE(I,J,K) - A real three dimensional array containing the s^2 coefficients of the second order factors; i.e., a_2 in (A-34).
- N1(I,J) - An integer array that denotes the number of cascaded first order factors for each subsystem.
- N2(I,J) - An integer array denoting the number of cascaded second order factors for each subsystem.
- DV(I) - A real one dimensional array representing the directional vector $\underline{d}$ in equation (A-5).
- ZETA - A real variable that denotes the desired minimum damping ratio.
- KKK - An integer used to count the number of elements in the directional vector.
- STEP - A real variable denoting the step size .
- PMG - A real variable denoting the magnitude of the partial vector.
- KRE - A program control integer.
- A,B,C,D,E - Parameter variables used to dimension the arrays by the maximum number of elements allowable.
- NRTR1, NRTR2 - An integer denoting whether first or second order factors, respectively, are constrained to the left half plane: if 1, the factors are unconstrained; otherwise, constrained.

Output Variables

The following variables are defined in the same manner as their respective input variables, but have been updated in the subprogram:

STEP, KKK, KRE, LPV.

APPENDIX B

FORTRAN IV LISTING OF THE COMPENSATOR IMPROVEMENT PROGRAM

This appendix contains a complete Fortran version of the Compensator Improvement Program for Multi-variable Control Systems. The program is completely self-contained, i.e., it requires no system library, etc. The necessary data input is explained in the comment statements preceding the main program. The subprograms are listed in alphabetical sequence. For information regarding subprogram theory or variables, refer to the respective synopsis of Appendix A.

THE MULTIVARIABLE COMPENSATOR IMPROVEMENT PROGRAM
 CREATED AT
 MISSISSIPPI STATE UNIVERSITY
 ELECTRICAL ENGINEERING DEPARTMENT
 BY
 L. L. GRESHAM
 J. R. MITCHELL
 AUGUST 1977

CIP MAIN PROGRAM

PARAMETER A=2, B=4, C=4, D=100, E=20, F=10*A*B*C, H=4, A2=2*A, C5=15*A*B*C, E2=2*E

PARAMETER EXPLANATION:

A COINCIDES WITH THE MAX NO. OF CONTROL INPUTS
 B COINCIDES WITH MAX NO. OF OUTPUTS
 C COINCIDES WITH THE MAX COMPENSATOR FACTORS
 D COINCIDES WITH MAX NO. OF FREQUENCY DATA POINTS
 E IS MAX NO. OF MARGIN POINTS ALLOWABLE.
 F IS 10*C
 H IS THE MAX. NO. OF FREQ. DEPENDENCE OF DESIGN SPECS.
 C5 IS 5*C
 E2 IS 2*E
 A2 IS 2*A

CIP DATA CARDS

CCARD NO.	VARIABLES READ	FORMAT
1	MODE,NZERO1,NZERO2,NPOLE1 NPOLE2,KEY,ZETAZ,ZETAP	A4,6X,5I5,2G10.5
2	ID	15A4
3	KSTART,KQUIT,KPOINT, KPRINT,KIN,KOUT	6I5
4	STPMAX,STPMIN,PINACT	3F10.5
5	KEYOUT(I),I=1,4	4I5
6	NONCTR(I),NOLRHP(I),I=1,KIN	6I5
7	NG,NP,NS,NA,NV	6I5
8	GMF(L),GMR(L),L=1,NG	8G10.3
.	.	.
.	.	.
9+NG	PMF(L),PMR(L),L=1,NP	8G10.3
.	.	.
.	.	.
10+NG+NP	SMF(L),SMR(L),L=1,NS	8G10.3
.	.	.
.	.	.

```

C 11+NG+NP+NS  ASF(L),ASF(L),L=1,NA      8G10.3
C  :
C  :
C  :
C 12+NG+NP+NS+NA F1,F2, . . . ,F10      8G10.3
C      OMEGA(K),G(I,J,K),J=1,KIN,
C      I=1,KOUT      4G20.5
C      GAIN(I,J),N1(I,J),N2(I,J),
C      M1(I,J),M2(I,J),KONT(I,J)      610.5,515
C      ZA(I,J,L),ZB(I,J,L),L=1,N1      7G10.5
C      ZC(I,J,L),ZD(I,J,L),ZE(I,J,L),
C      L=1,N2      7G10.5
C      PA(I,J,L),PB(I,J,L),L=1,M1      7G10.5
C      PC(I,J,L),PD(I,J,L),PE(I,J,L),
C      L=1,M2      7G10.5
C
C DEFINITIONS OF I/O VARIABLES
C
C ITERATION CONTROL DATA
C
C MODE -DETERMINES WHETHER THE PROGRAM IS TO OPERATE IN THE
C SUM IMPROVEMENT FREQUENCY MODE (SIFR) OR THE TOTAL
C IMPROVEMENT FREQUENCY MODE (TIFR). IF DATA CARD
C BLANK THE PROGRAM AUTOMATICALLY DEFAULTS TO THE
C TIFR MODE. IF SIFR IS DESIRED THEN MODE=SIFR.
C NZERO1-DETERMINES WHETHER 1-ST ORDER ZERO FACTORS ARE
C CONSTRAINED TO L.H.P. (IF .EQ. TO 1, UNCONSTRAINED
C OTHERWISE, CONSTRAINED).
C NZERO2-SAME AS NZERO1, EXCEPT FOR 2-ND ORDER FACTORS.
C NPOLE1-SAME AS NZERO1, EXCEPT FOR 1-ST ORDER POLE FACTORS.
C NPOLE2-SAME AS NZERO1 EXCEPT FOR 2-ND ORDER POLE FACTORS.
C ZETAZ- MINIMUM DAMPING RATIOS FOR COMPENSATOR ZEROS; APPLIES
C ONLY IF NZERO .NE.1.
C KEY = 0 , NYQUIST SUBPROGRAM NOT CALLED
C ZETAP - SAME AS ZETAZ APPLIED TO POLES
C KSTART -STARTING ITERATION NO.
C KQUIT - STOPPING ITERATION NUMBER
C KPOINT -NO. OF POINTS FROM OPEN LOOP FREQ. RESPONSE USED
C IF KPOINT=0, READ A FREQUENCY, THEN EACH CHANNEL'S
C STPMAX -MAXIMUM CHANGE TO BE MADE IN COMPENSATOR COEFFICIENTS
C SMALLEST COMPENSATOR COEFFICIENT OF THE INITIAL
C COMPENSATOR)
C STPMIN - MINIMUM STEP SIZE DESIGNATOR
C PINACT -LARGEST DIFFERENCE BETWEEN A CONSTRAINT AND ITS
C DESIRED VALUE IN GOING FROM INACTIVITY TO ACTIVITY
C KPRINT - NO. OF ITERATIONS SKIPPED BETWEEN PRINTING OF INFOR.
C
C KEYOUT=1 , REQUESTS PRINTOUT
C KEYOUT=0 , NO PRINTOUT
C KEYOUT(1) - COMPLETE ITERATION OUTPUT
C KEYOUT(2) - OUTPUT COMPENSATOR INFORMATION
C KEYOUT(3) - OUTPUT FREQUENCY RESPONSE
C KEYOUT(4) - OUTPUT MARGIN SPECIFICATIONS
C NONCTR(I) - NUMBER OF POLES ON CONTOUR
C NOLRHP(I) - NUMBER OF POLES INSIDE CONTOUR

```

```
C  
C      VARIABLES FOR MARGIN RADII SPECIFICATIONS  
C  
C      NUMBER OF MARGIN RADII TO BE SPECIFIED:  
C      NG - NO. OF GAIN MARGIN RADII SPECIFIED  
C      NP - NO. OF PHASE MARGIN RADII SPECIFIED  
C      NS - NO. OF STABILITY MARGIN RADII SPECIFIED  
C      NA - NO. OF ATTENUATION MARGIN RADII SPECIFIED  
C  
C      VARIABLES FOR GAIN MARGIN RADII DESIGNATIONS  
C      IF FREQ .GT. GMF(I) BUT .LT. GMF(I+1)   DESIRED MARGIN = GMR(I)  
C      IF FREQ .GT. GMF(I+1) BUT .LT. GMF(I+2) DESIRED MARGIN = GMR(I+1)  
C      ETC.  
C      IF FREQ .GT. GMF(NF)                     DESIRED MARGIN = GMR(NF)  
C  
C      VARIABLES FOR PHASE MARGIN RADII DESIGNATIONS  
C      IF FREQ .GT. PMF(I) BUT .LT. PMF(I+1)   DESIRED MARGIN = PMR(I)  
C      IF FREQ .GT. PMF(I+1) BUT .LT. PMF(I+2) DESIRED MARGIN = PMR(I+1)  
C      ETC.  
C      IF FREQ .GT. PMF(NF)                     DESIRED MARGIN = PMR(NF)  
C      VARIABLES FOR STABILITY MARGIN RADII DESIGNATIONS  
C      IF FREQ .GT. SMF(I) BUT .LT. SMF(I+1)   DESIRED MARGIN = SMR(I)  
C      IF FREQ .GT. SMF(I+1) BUT .LT. SMF(I+2) DESIRED MARGIN = SMR(I+1)  
C      ETC.  
C      IF FREQ .GT. SMF(NF)                     DESIRED MARGIN = SMR(NF)  
C  
C      VARIABLES FOR ATTENUATION MARGIN RADII DESIGNATIONS  
C      IF FREQ .FT. ASF(I) BUT .LT. ASF(I+1)   DESIRED MARGIN = ASR(I)  
C      IF(FREQ .GT. ASF(I+1) BUT .LT. ASF(I+2)) DESIRED MARGIN = ASR(I)  
C      ETC.  
C      IF FREQ .GT. ASF(NA)                    DESIRED MARGIN = ASR(NA)  
C  
  
C      F1 : F2 - FREQUENCIES BETWEEN WHICH G.M.'S ARE FOUND  
C      F3 : F4 - FREQUENCIES BETWEEN WHICH P.M.'S ARE FOUND  
C      F5 : F6 - FREQUENCIES BETWEEN WHICH S.M.'S ARE FOUND  
C      F7 : F8 - FREQUENCIES BETWEEN WHICH A.M.'S ARE FOUND  
C  
  
C      VARIABLES FOR PLANT DYNAMICS  
C  
C      KIN IS NO. OF CONTROL INPUTS.  
C      KOUT IS NO. OF OUTPUTS  
C      OMEGA(I) - ITH FREQ.(ASSUMED TO BE IN HZ.)  
C      G(I,J,K) - PLANT DYNAMICS WITH INDEX F(KIN,KOUT,KPOINT)  
C  
C      DESCRIPTION OF COMPENSATION  
C  
C      GAIN(I)-DENOTES INITIAL D. C. GAIN VALUE FOR I-TH CHANNEL  
C      KONT(I)-D.C. DESIGNATOR FOR I-TH CHANNEL  
C          KONT(I)=1    GAIN ALLOWED TO VARY  
C          KONT(I)=2    GAIN NOT ALLOWED TO VARY
```

```

C      N1(I,J) - NO. OF 1ST ORDER NUMERATOR COEFFS. OF COMPENSATOR (I,J)
C      N2(I,J) - NO. OF 2ND ORDER NUMERATOR COEFFS. OF COMPENSATOR (I,J)
C      M1(I,J) - NO. OF 1ST ORDER DENOMINA. COEFFS. OF COMPENSATOR (I,J)
C      M2(I,J) - NO. OF 2ND ORDER DENOMINA. COEFFS. OF COMPENSATOR (I,J)
C      KTH CASCADE COMPENSATOR(I,J,K) COEFFICIENTS WHERE I=KIN, J=KOUT
C      FIRST ORDER NUMERATOR FACTOR COEFFICIENTS: (ZA + ZB*S)
C      ZA(I,J,K) - 1ST ORDER NUM. FACTOR CONSTANT OF KTH CASCADE COMP.
C      ZB(I,J,K) - 1ST ORDER NUMERATOR FACTOR COEFFICIENT(ZB*S) OF KTH
C      SECOND ORDER NUMERATOR FACTOR COEFFICIENTS: (ZC + ZD*S + ZE*S**2)
C      ZC(I,J,K) - 2ND ORDER NUM. FACTOR CONSTANT OF KTH CASCADE COMP.
C      ZD(I,J,K) - COEFFICIENT ZD*S OF KTH CASCADE COMPENSATOR
C      ZE(I,J,K) - COEFFICIENT ZE*S**2 OF KTH CASCADE COMPENSATOR
C      FIRST ORDER DENOMINATOR FACTOR COEFFICIENTS: (PA + PB*S)
C      PA(I,J,K) - 1ST ORDER DENOMINATOR FACTOR CONSTANT
C      PB(I,J,K) - 1ST ORDER DENOMINATOR FACTOR COEFF. (PB*S) OF KTH
C      SECOND ORDER DENOMINATOR FACTOR COEFFICIENTS: (PC+PD*S+PE*S**2)
C      PC(I,J,K) - 2ND ORDER DENOM. FACTOR CONSTANT OF KTH CASC. COMP.
C      PD(I,J,K) - COEFFICIENT PD*S OF KTH CASCADE COMPENSATOR
C      PE(I,J,K) - COEFFICIENT PE*S**2 OF KTH CASCADE COMPENSATOR
C      INTEGER TYPE, TYACT, ACTIVE, ACTDES, XXP, XXG, XXS, XXA
C      COMPLEX G(B,A,D), GC(A,B,D), T(A,A,D), OMEGA(D), CT(D,A), PCG, C1(A,A), C
1 I(A,A), WORKI(A,A2)
C      DIMENSION KONT(A,B), GAIN(A,B), N1(A,B), N2(A,B), M1(A,B), M2(A,B)
C      1, ZA(A,B,C), ZB(A,B,C), ZC(A,B,C), ZD(A,B,C), ZE(A,B,C), PA(A,B,C)
C      2, PB(A,B,C), PC(A,B,C), PD(A,B,C), PE(A,B,C), GMF(H), GMR(H), PMF(
C      3H), PMR(H), SMF(H), SMR(H), ASF(H), ASR(H), NM(A), KOLD(D), PG(E,F
C      4), DV(F), WEIGHT(E), KMIN(A), SML(E,A), TYACT(E), RQ(E,A), TYPE(E,
C      5A), ACTIVE(E,A), STBM(E,A), KPTS(E,A), NIT(A), KINACT(E,A), NML(A)
C      6, KACT(E,A), ID(15), ACTDES(2), PFX(E,E), PFY(E,E), WORKR(E,E2), K
C      7NEW(D), NONCTR(A), NOLRHP(A), KEYOUT(4)
C      COMMON ITER, KSTART
C      DATA IBLANK, XXP, XXG, XXS, XXA /4H ,1HP,1HG,1HS,1HA/ ACTDES(1), ACT
C      1DES(2), ITIFR /3HYES, 2HNO, 4HTIFR/
C      KPTMAX=D
C      DATA INPUT BLOCK . . . . .
C      ITERATION CONTROL DATA . . . . .
C      READ (5,10) MODE, NZERO1, NZERO2, NPOLE1, NPOLE2, KEY, ZETAZ, ZETAP, ID
10  FORMAT (A4, 6X, SI5, 2G10.5/15A4)
C      WRITE (6,20) ID
20  FORMAT (1H1, 10X, 15A4)
C      WRITE (6,30) NZERO1, NZERO2, NPOLE1, NPOLE2
30  FORMAT ('0', 5X, 'NZERO1=', I2, ' NZERO2=', I2, ' NPOLE1=', I2, ' NPOLE2
C      1=', I2)
C      IF (MODE.EQ.IBLANK) MODE=ITIFR
C      WRITE (6,40) MODE
40  FORMAT ('0', 5X, 'THE PROGRAM IS IN THE ', A4, ' MODE')
C      READ (5,50) KSTART, KQUIT, KPOINT, KPRINT, KIN, KOUT, STPMAX, STPMIN, PINA
C      1CT
50  FORMAT (6I5/3F10.5)
C      READ (5,60) (KEYOUT(I), I=1,4)
C      READ (5,60) (NONCTR(I), NOLRHP(I), I=1, KIN)
60  FORMAT (6I5)

```

```

C   DESIGN SPECIFICATION DATA . . . . .
    READ (5,60) NG,NP,NS,NA,NV
    WRITE (6,80) KSTART,KQUIT,KPOINT,KPRINT
C   READ MARGIN FREQUENCIES IN ASCENDING ORDER
    READ (5,70) (GMF(K),GMR(K),K=1,NG)
    READ (5,70) (PMF(L),PMR(L),L=1,NP)
    READ (5,70) (SMF(K),SMR(K),K=1,NS)
    READ (5,70) (ASF(L),ASR(L),L=1,NA)
    READ (5,70) F1,F2,F3,F4,F5,F6,F7,F8,F9,F10
70   FORMAT (8G10,3)
80   FORMAT ('0',5X,'START ITER = ',I4,' STOP ITER = ',I4,2X,'NO. OF
1   IFREQ. POINTS = ',I5,2X,'PRINT INCREMENT = ',I5)
    WRITE (6,90) KIN,KOUT
90   FORMAT ('0',5X,'NUMBER OF CONTROL INPUT CHANNELS=',I5,5X,' NUMBER
1   OUTPUT CHANNELS = ',I5,/)
    WRITE (6,100) STPMAX,STPMIN,ZETAZ,ZETAP,PINACT
100  FORMAT ('0',5X,'MAXIMUM DESIGNATED STEP SIZE = ',F10.5,6X,'MINIMUM
1   DESIGNATED STEP SIZE = ',G10.5/6X,'MINIMUM COMPENSATOR ZEROS ZETA
2   = ',G10.5/6X,'MINIMUM COMPENSATOR POLES ZETA = ',G10.5/6X,'AMT. AB
3  OVE SPECS. TO BE KEPT IN ACT.=',F10.5)
    DO 110 I=1,KIN
110  WRITE (6,120) I,NONCTR(I),NOLRHP(I)
120  FORMAT ('0',5X,'FOR SYSTEM NO.',I3,'; NONCTR=',I3,' AND NOLRHP=',I
13)
    WRITE (6,130) NG,NP,NS,NA
130  FORMAT ('0',5X,'NG=',I2,' ,NP=',I2,' ,NS=',I2,' ,NA=',I2)
    WRITE (6,140)
140  FORMAT ('0',5X,'DESIRED MARGIN RADII DESIGNATIONS',///6X,'DESIRED
1   GAIN MARGIN DESIGN SPECIFICATIONS')
    WRITE (6,150) (GMF(K),GMR(K),K=1,NG)
150  FORMAT (/,15X,'IF FREQUENCY .GE.',F10.5,5X,'DESIRED MARGIN IS',F10
1.5)
    WRITE (6,160) F1,F2
160  FORMAT (' ',5X,'GAIN MARGINS ARE DETERMINED BETWEEN THE FREQUENCIE
1   S OF',F10.5,2X,'AND',F10.5)
    WRITE (6,170)
170  FORMAT ('0',5X,'DESIRED PHASE MARGIN DESIGN SPECIFICATIONS')
    WRITE (6,180) (PMF(L),PMR(L),L=1,NP)
    WRITE (6,180) F3,F4
180  FORMAT (' ',5X,'PHASE MARGINS ARE DETERMINED BETWEEN THE FREQUENCIE
1   S OF',F10.5,2X,'AND',F10.5)
    WRITE (6,190)
190  FORMAT ('0',5X,'DESIRED STABILITY MARGIN DESIGN SPECIFICATIONS')
    WRITE (6,200) (SMF(K),SMR(K),K=1,NS)
    WRITE (6,200) F5,F6
200  FORMAT (' ',5X,'STABILITY MARGINS ARE FOUND BETWEEN THE FREQUENCIE
1   S OF',F10.5,2X,'AND',F10.5)
    WRITE (6,210)
210  FORMAT ('0',5X,'DESIRED ATTENUATION MARGIN DESIGN SPECIFICATIONS')
    WRITE (6,220) (ASF(K),ASR(K),K=1,NA)
    WRITE (6,220) F7,F8
220  FORMAT (' ',5X,'ATTENUATION MARGINS ARE FOUND BETWEEN THE FREQUENCIE
1   S OF',F10.5,2X,'AND',F10.5)
C   READ DATA ON O.L. SYSTEM . . . . .
C   I,J,K ALWAYS DENOTE COUNTER KIN,KOUT,KOATA POINT RESPECTIVELY

```

```

WRITE (6,230)
230  FORMAT (5X,"OPEN LOOP SYSTEM INPUT FREQUENCY RESPONSE: ",/,31X,"CO
      1MPLEX OMEGA",13X,"COMPLEX G(J,W)"/,5X,"DATA",3X,"CHANNEL(I,J)",5X,"
      2REAL(W)",5X,"IMAG(W)",5X,"REAL(G)",5X,"IMAG(G)"/,/)
      DO 260 K=1,KPOINT
        READ (5,240) OMEGA(K),((G(I,J,K),J=1,KIN),I=1,KOUT)
240  FORMAT (4G20.5)
      DO 260 I=1,KOUT
        DO 260 J=1,KIN
          WRITE (6,250) K,I,J,OMEGA(K),G(I,J,K)
250  FORMAT (5X,I3,2(5X,I2),2(5X,2G10.4))
260  CONTINUE
C    COMPENSATOR INPUT DATA . . . . .
      DO 320 I=1,KIN
        DO 320 J=1,KOUT
          READ (5,270) GAIN(I,J),N1(I,J),N2(I,J),M1(I,J),M2(I,J),KONT(I,J)
270  FORMAT (1G10.5,5I5)
C    READ COEFFICIENTS IN ASCENDING POWERS OF S, IN 1ST AND 2ND ORDER
      NC=N1(I,J)
      IF (NC.EQ.0) GO TO 280
      READ (5,310) (ZA(I,J,L),ZB(I,J,L),L=1,NC)
280  NC=N2(I,J)
      IF (NC.EQ.0) GO TO 290
      READ (5,310) (ZC(I,J,L),ZD(I,J,L),ZE(I,J,L),L=1,NC)
290  MC=M1(I,J)
      IF (MC.EQ.0) GO TO 300
      READ (5,310) (PA(I,J,L),PB(I,J,L),L=1,MC)
300  MC=M2(I,J)
      IF (MC.EQ.0) GO TO 320
      READ (5,310) (PC(I,J,L),PD(I,J,L),PE(I,J,L),L=1,MC)
310  FORMAT (7G10.5)
320  CONTINUE
C    DATA INPUT COMPLETED . . . . .
C    DETERMINE NO. OF INDEPENDENT PARAMETERS
      KNOT=0
      LNOT=0
      KVAR=0
      DO 330 I=1,KIN
        DO 330 J=1,KOUT
          KNOT=KNOT+2*N1(I,J)+3*N2(I,J)
          LNOT=LNOT+2*M1(I,J)+3*M2(I,J)
          KVAR=KVAR+N1(I,J)+N2(I,J)*2
          KVAR=KVAR+M1(I,J)+M2(I,J)*2
330  IF (KONT(I,J).EQ.1) KVAR=KVAR+1
      NPARC=KNOT+LNOT
      ITER=KSTART
      STEP=STPMAX
      STPOLD=STPMAX
      KPOLD=KPOINT
      DO 340 I=1,KPOLD
340  KOLD(I)=I
      KPR=-1
      DATA RAD2 /114.591559/
      DO 360 I=1,NP
        IF ((PMR(I).GE.0.).AND.(PMR(I).LE.180.)) GO TO 360
        WRITE (6,350)
350  FORMAT ('0',5X,"**** HEY DUMMY  YOU HAVE MADE A MISTAKE" ON THE
      1PHASE MARGIN SPECIFICATIONS ****")

```

```

      STOP
360  PMR(I)=2.*SIN(PMR(I)/RAD2)
      FNYQT=AMAX1(GMF(NG),PMF(NP),SMF(NS))
370  KRESET=0
      IF ((ITER.EQ.KSTART).OR.(KPOINT.LT.1.5*KSTAND)) GO TO 380
      CALL DELETE (KPOINT,KIN,KOUT,OMEGA,G,NIT,KINACT,NML,KACT,ITER,KPOL
1D,KOLD,KNEW,A,B,C,D,E)
      KRESET=1
C    CALCULATION OF COMPENSATED FREQUENCY RESPONSE . . . . .
380  DO 390 K=1,KPOINT
390  CALL EVAL (OMEGA(K),GC,G,T,K,KIN,KOUT,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,
1E,N1,N2,M1,M2,GAIN,KONT,A,B,C,D)
      IF (KPR.EQ.KPRINT) KPR=0
      KPR=KPR+1
      IF (ITER.NE.KSTART) GO TO 400
      IF (KEYOUT(2).EQ.1) CALL OUTPT (0,KSYS,KPOINT,CT(1,KSYS),OMEGA,ITE
1R,N,KPTS(1,KSYS),KMIN,STBM(1,KSYS),RQ,TYPE(1,KSYS),ACTIVE,KIN,
2 KOUT,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N2,M1,M2,GAIN,KONT,A,B,C,K
3OLD)
      IF (ITER.GT.KQUIT) GO TO 1020
C    DETERMINATION - OPEN LOOP FREQUENCY RESPONSE CT OF KSYM . . . . .
400  NAM=0
      NSUM=0
      MAD=0
      LPV=KVARY
      ADD=1.E-15
      DO 750 KSYM=1,KIN
      KPLAST=KPOINT
      DO 410 K=1,KPOINT
      CALL CRT (1,K,KSYS,T(1,1,K),KIN,KOUT,CT(1,KSYS),G(1,1,K),PCG,I1,J1
1,C1,C1,WORKI,A,B,D)
      IF (KEYOUT(3).EQ.1) CALL OUTPT (1,KSYS,KPOINT,CT(1,KSYS),OMEGA,ITE
1R,N,KPTS(1,KSYS),KMIN,STBM(1,KSYS),RQ,TYPE(1,KSYS),ACTIVE,KIN,
2 KOUT,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N2,M1,M2,GAIN,KONT,A,B,C,K
3OLD)
410  CONTINUE
C    DETERMINATION OF MARGINS . . . . .
420  NM(KSYM)=0
C    DETERMINATION OF GAIN MARGINS BETWEEN F1 AND F2:
      CALL GAINMG (CT(1,KSYS),KPOINT,NM(KSYM),F1,F2,KPTS(1,KSYS),STBM(1,
1KSYS),OMEGA)
      KPM=NM(KSYM)+1
      NGMS=NM(KSYM)
      IF (NM(KSYM).EQ.0) GO TO 440
      CALL ADDPTS (KPOINT,KIN,KOUT,1,NM(1),STBM(1,KSYS),KPTS(1,1),CT(1,1
1),G,GC,T,OMEGA,KGOBAK,KPTMAX,NIT,KINACT,NML,KACT,T(1,1),KPOLD,KOLD,K
2SYM,C1,C1,WORKI,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N2,M1,M2,GAIN,KON
3T,A,B,C,D,E)
      IF (KPOINT.GT.KPTMAX) GO TO 450
      IF (KGOBAK.EQ.1) GO TO 420
      N=NM(KSYM)
C    SETTING DESIRED STABILITY RADII OF GM'S:
      DO 430 I=1,N
      TYPE(I,KSYS)=XXG
      KWHICH=KPTS(I,KSYS)
      FREHZ=AIMAG(OMEGA(KWHICH))

```

```

DO 430 L=1,NG
  IF (FREQZ.GE.GMF(L)) RQ(I,KSYP)=GMR(L)
430  CONTINUE
440  NM(KSYM)=NGMS
C    DETERMINATION OF PHASE MARGINS BETWEEN F3 AND F4:
    CALL PHASEM (CT(1,KSYP),KPOINT,NM(KSYM),F3,F4,KPTS(1,KSYP),STBM(1,
1KSYP),OMEGA)
    N=NM(KSYM)
    IF (N.LT.KPM) GO TO 490
    CALL ADDPTS (KPOINT,KIN,KOUT,KPM,NM(1),STBM(1,KSYP),KPTS(1,1),CT(1
1,1),G,GC,T,OMEGA,KGOBAK,KPTMAX,NIT,KINACT,NML,KACT(1,1),KPOLD,KOLD
2,KSYP,C1,C1,WORKI,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N2,M1,M2,GAIN,K
3ONT,A,B,C,D,E)
    IF (KPOINT.LE.KPTMAX) GO TO 470
450  WRITE (6,460) KPTMAX
460  FORMAT ('0',5X,3(1H*),2X,'TERMINATION REASON: NO. OF FREQ. "POIN
1TS HAS EXCEEDED",15,2X,3(1H*))
    GO TO 900
470  IF (KGOBAK.EQ.1) GO TO 440
    N=NM(KSYM)
C    SETTING DESIRED STABILITY RADII OF P.M.'S:
    DO 480 I=KPM,N
      TYPE(I,KSYP)=XXP
      KWHICH=KPTS(I,KSYP)
      FREQZ=AIMAG(OMEGA(KWHICH))
      DO 480 L=1,NP
        IF (FREQZ.GE.PMF(L)) RQ(I,KSYP)=PMR(L)
480  CONTINUE
      KPM=N+1
490  CONTINUE
      KLAST=NM(KSYM)
500  NM(KSYM)=KLAST
      KSTBM=KPM
C    DETERMINATION OF STABILITY MARGINS
      CALL SRMINS (CT(1,KSYP),KPOINT,NM(KSYM),1.,1,F5,F6,KPTS(1,KSYP),ST
1BM(1,KSYP),OMEGA)
      N=NM(KSYM)
      IF (N.LT.KPM) GO TO 590
      CALL ADDPTS (KPOINT,KIN,KOUT,KPM,NM(1),STBM(1,KSYP),KPTS(1,1),CT(1
1,1),G,GC,T,OMEGA,KGOBAK,KPTMAX,NIT,KINACT,NML,KACT(1,1),KPOLD,KOLD
2,KSYP,C1,C1,WORKI,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N2,M1,M2,GAIN,K
3ONT,A,B,C,D,E)
      IF (KPOINT.GT.KPTMAX) GO TO 450
      IF (KGOBAK.EQ.1) GO TO 500
      IF (ITER.EQ.KSTART) GO TO 510
      IF (KEY.EQ.0) GO TO 530
510  CALL NYQIST (KPOINT,CT(1,KSYP),NOLRHP(KSYM),NONCTR(KSYM),NCIRL,NZ,
1FNYQT,OMEGA)
      IF (NZ.EQ.0) GO TO 530
      IF (ITER.NE.KSTART) GO TO 760
      WRITE (6,520) KSYM
520  FORMAT ('0',5X,'WITH THE INITIAL COMPENSATION FOR SYSTEM NO.',I3,1
1X,'HAS CLOSED LOOP POLES INSIDE THE CONTOUR IN THE'' PHASE STABILI
ZZATION REGION')
530  CONTINUE
      IF (KPR+1.NE.KPRINT) GO TO 530

```



```

C      SETTING DESIRED STABILITY MARGINS
      N=NM(KSYM)
      DO 540 I=KPM,N
      TYPE(I,KSYM)=XXS
      KWHICH=KPTS(I,KSYM)
      FREHZ=AIMAG(OMEGA(KWHICH))
      DO 540 L=1,NS
      IF (FREHZ.GE.SMF(L)) RQ(I,KSYM)=SMR(L)
540  CONTINUE
C      CHECK TO SEE IF ANY P.M.'S,G.M.'S, OR S.M.'S EQUAL
C      IF THERE RESULTS SOME THAT ARE EQUAL ONLY THE FIRST IS RETAINED.
      IF (KSTBM.LE.1) GO TO 580
      DO 570 LB=1,N
      NSG=LB+1
550  CONTINUE
      IF (NSG.GT.NM(KSYM)) GO TO 580
      DO 570 I=NSG,N
      IF (KPTS(LB,KSYM).NE.KPTS(I,KSYM)) GO TO 570
      N=N+1
      DO 560 L=LB,N
      KPTS(L,KSYM)=KPTS(L+1,KSYM)
      STBM(L,KSYM)=STBM(L+1,KSYM)
      RQ(L,KSYM)=RQ(L+1,KSYM)
560  TYPE(L,KSYM)=TYPE(L+1,KSYM)
      GO TO 550
570  CONTINUE
580  CONTINUE
      KPM=N+1
590  CONTINUE
      KPADD=KPOINT-KPLAST
      IF (KPADD.NE.0) WRITE (6,600) KPADD,ITER,KPOINT
600  FORMAT (1H0,5X,I3,1X,34HPOINTS WERE ADDED ON ITERATION NO.,I4,/10X
1,26HNO. OF FREQ. POINTS IS NOW,I4)
      KMIN(KSYM)=N
      IF ((ITER.EQ.KSTART).OR.(KRESET.EQ.1)) KSTAND=KPOINT
C      DETERMINATION OF ATTENUATION MARGINS
      CALL SRMINS (CT(1,KSYM),KPOINT,NM(KSYM),0.,-1,F7,F8,KPTS(1,KSYM),S
1TBM(1,KSYM),OMEGA)
C      SETTING DESIRED STABILITY MARGINS . . . . .
      N=NM(KSYM)
      IF (N.LT.KPM) GO TO 620
      DO 610 I=KPM,N
      TYPE(I,KSYM)=XXA
      KWHICH=KPTS(I,KSYM)
      FREHZ=AIMAG(OMEGA(KWHICH))
      DO 610 L=1,NA
      IF (FREHZ.GE.ASF(L)) RQ(I,KSYM)=ASR(L)
610  CONTINUE
620  CONTINUE
C      CHECKING MODE REQUIREMENTS . . . . .
      IF (ITER.EQ.KSTART) GO TO 750
      PORM=1.
      N=NM(KSYM)
      NI=NIT(KSYM)
      NL=NML(KSYM)
      IF (N.EQ.0) GO TO 750
      KCHMIN=MIN0(N,NL+NI)
      KKK=0

```

```

DO 660 I=1,N
DO 630 J=1,NL
DO 630 K=-2,2
630 IF (KPTS(I,KSVM).EQ.KACT(J,KSVM)+K) GO TO 650
DO 640 J=1,NI
DO 640 K=-2,2
640 IF (KPTS(I,KSVM).EQ.KINACT(J,KSVM)+K) GO TO 650
GO TO 660
650 KKK=KKK+1
660 CONTINUE
IF (KKK.LT.KCHMIN) GO TO 760
NSUM=NSUM+N
DO 740 I=1,N
IF (I.GT.KMIN(KSYM)) PORM=-1.
IF (PORM*(STBM(I,KSVM)-RQ(I,KSVM)).GE.0.) GO TO 740
IF (NIT(KSYM).EQ.0) GO TO 700
DO 690 J=1,NI
DO 670 K=-2,2
670 IF (KPTS(I,KSVM).EQ.KINACT(J,KSVM)+K) GO TO 680
GO TO 690
680 IF (PORM*(RQ(I,KSVM)-STBM(I,KSVM)).GT.PINACT) GO TO 760
690 CONTINUE
700 IF (NL.EQ.0) GO TO 740
DO 710 J=1,NL
DO 710 K=-2,2
710 IF (KPTS(I,KSVM).EQ.KACT(J,KSVM)+K) GO TO 720
MAD=MAD+1
GO TO 740
720 CONTINUE
IF (MODE.NE.ITIFR) GO TO 730
IF (PORM*(STBM(I,KSVM)-SML(J,KSVM)).LT.-1.E-06) GO TO 760
730 CONTINUE
ADD=ADD+PORM*(STBM(I,KSVM)-SML(J,KSVM))
740 CONTINUE
750 CONTINUE
IF (ITER.EQ.KSTART) GO TO 800
IF (MAD.EQ.NSUM) ADD=1.
IF (ADD.GT.0.) GO TO 770
760 STEP=-ABS(STEP)/2.
IF (ABS(STEP).LT.STPMIN) GO TO 780
ITER=ITER+1
KPR=KPR+1
GO TO 980
770 STEP=1.41416*ABS(STPOLD)
IF (ABS(STEP).GT.STPMAX) STEP=STPMAX
GO TO 800
C OUTPUT CONTROL
780 WRITE (6,790) STPMIN
790 FORMAT ('0',5X,'**** TERMINATION - STEP SIZE LESS THAN ',G15.5,2X,
1'*****')
GO TO 900
800 NAM1=0
IF ((KPR.EQ.KPRINT).OR.(ITER.EQ.KSTART).OR.(ITER.EQ.KQUIT)) WRITE
1(6,810) STEP
810 FORMAT ('0',25X,'PRESENT STEP SIZE =',G15.5)
DO 840 KSYM=1,KIN
N=NM(KSYM)
KSM=KMIN(KSYM)

```

```

C      DETERMINING ACTIVE MARGINS . . . . .
      NIT(KSYM)=0
      K=0
      DO 830 I=1,N
      PORM=1.
      IF (I.GT.KMIN(KSYM)) PORM=-1.
      IF (PORM*STBM(I,KSYM).LT.PORM*RQ(I,KSYM)) NAM1=1
      IF (PORM*STBM(I,KSYM).GT.PORM*RQ(I,KSYM)+PINACT) GO TO 820 .
C      ACTIVE CONSTRAINTS
C      TYACT IS NO. OF TYPES ACTIVE
C      KACT IS POINTER TO ACTIVES
      K=K+1
      TYACT(K)=TYPE(I,KSYM)
      KACT(K,KSYM)=KPTS(I,KSYM)
      SML(K,KSYM)=STBM(I,KSYM)
      ACTIVE(I,KSYM)=ACTDES(1)
      GO TO 830
C      INACTIVE CONSTRAINTS
820    NIT(KSYM)=NIT(KSYM)+1
      NQ=NIT(KSYM)
      KINACT(NQ,KSYM)=KPTS(I,KSYM)
      ACTIVE(I,KSYM)=ACTDES(2)
830    CONTINUE
      IF ((KEYOUT(4).EQ.1).AND.((KPR.EQ.KPRINT).OR.(ITER.EQ.KSTART).OR.(
1 ITER.EQ.KQUIT))) CALL OUTPT (2,KSYM,KPOINT,CT(1,KSYM),OMEGA,ITER,N
2 ,KPTS(1,KSYM),KSM,STBM(1,KSYM),RQ(1,KSYM),TYPE(1,KSYM),ACTIVE(1,K
3 YM),KIN,KOUT,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N2,M1,M2,GAIN,KONT,A
4 ,B,C,KOLD)
      NML(KSYM)=K
      CALL PARTIAL (OMEGA,NML(KSYM),CT(1,KSYM),KACT(1,KSYM),TYACT(1),T,G,
1 GC,KSYM,C1,CI,WORKI,A,B,C,D,E,F,CS,PG(NAM+1,1),PG(NAM+1,KNOT+1),KI
2 N,KOUT,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N2,M1,M2,GAIN,KONT)
      NAM=NAM+NML(KSYM)
840    CONTINUE
C      NORMALIZING PG
      DO 860 I=1,NAM
      SUM=0.
      DO 850 J=1,NPARC
850    SUM=SUM+PG(I,J)**2
      SUM=DSQRT(SUM)
      DO 860 J=1,NPARC
      IF (SUM.LT.0.1E-06) SUM=1.E-06
860    PG(I,J)=PG(I,J)/SUM
      IF (ITER.GE.KQUIT) WRITE (6,870)
870    FORMAT ('0',5X,'*** TERMINATION REASON:  MAXIMUM ITERATIONS ***')
      IF (ITER.GE.KQUIT) GO TO 900
      IF (NAM1.NE.0) GO TO 920
      WRITE (6,880)
880    FORMAT ('0',15X,'**** ALL SYSTEM REQUIREMENTS HAVE BEEN MET ****')
      WRITE (6,890) ITER
890    FORMAT ('0',5X,'THE SUBOPTIMAL COMPENSATOR OCCURRED ON ITERATION N
10.: ',I3,'/5X,'WITH THE FOLLOWING COEFFICIENTS:  '/')
900    DO 910 KSYM=1,KIN
      N=NM(KSYM)
910    IF (KEYOUT(1).EQ.1) CALL OUTPT (3,KSYM,KPOINT,CT(1,KSYM),OMEGA,ITE
1 R,N,KPTS(1,KSYM),KMIN(KSYM),STBM(1,KSYM),RQ(1,KSYM),TYPE(1,KSYM),A
2 CTIVE(1,KSYM),KIN,KOUT,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N2,M1,M2,G
3 AIN,KONT,A,B,C,KOLD)

```

```

      STOP
C     SET DOT PRODUCT
920   DO 930 N=1,NAM
930   WEIGHT(N)=1.
940   IF (NAM.LE.LPV) GO TO 960
      WRITE (6,950)
950   FORMAT ('0',5X,'***** TERMINATION REASON - NO. OF ACTIVE CONSTRAI
1NTS EXCEEDS THE NO. OF ALLOWABLE VARIRABLES *****')
      STOP
960   CALL DIRVEC (PG,NAM,NPARC,DV,WEIGHT,PFX,PFY,WORKR,E,F,C5,E2)
      KRE=0
      KKK=0
      CALL XCHECK (KIN,KOUT,ZA,ZB,ZC,ZD,ZE,N1,N2,DV,ZETAZ,PG,NAM,KRE,NZE
1RO1,NZERO2,LPV,A,B,C,D,E,F,KKK)
      IF ((KRE.EQ.3).AND.(ITER.EQ.KSTART)) GO TO 1000
      IF (KRE.EQ.1) GO TO 940
      CALL XCHECK (KIN,KOUT,PA,PB,PC,PD,PE,M1,M2,DV,ZETAP,PG,NAM,KRE,NP
1LE1,NPOLE2,LPV,A,B,C,D,E,F,KKK)
      IF ((KRE.EQ.3).AND.(ITER.EQ.KSTART)) GO TO 1000
      IF (KRE.EQ.1) GO TO 940
      PSQ=0.
      DO 970 I=1,NPARC
970   PSQ=PSQ+DV(I)**2
      PMG=SQRT(PSQ)
      IF (PMG.LT.1.E-08) PMG=1.E-08
980   DEL=STEP/PMG
      KKK=1
      CALL CHANGE (ZA,ZB,ZC,ZD,ZE,N1,N2,DV,DEL,KKK,KIN,KOUT,A,B,C)
      CALL CHANGE (PA,PB,PC,PD,PE,M1,M2,DV,DEL,KKK,KIN,KOUT,A,B,C)
      IF (KRE.NE.3) STPOLD=STEP
      IF (KRE.EQ.3) STEP=STEP+STPOLD
      KKK=0
      IF (KRE.EQ.3) GO TO 990
      KRE=0
      CALL YCHECK (KIN,KOUT,N1,N2,ZA,ZB,ZC,ZD,ZE,KRE,STPOLD,PMG,ZETAZ,NZ
1ERO1,NZERO2,DV,A,B,C,D,E,KKK,STEP)
      CALL YCHECK (KIN,KOUT,M1,M2,PA,PB,PC,PD,PE,KRE,STPOLD,PMG,ZETAP,NP
1OLE1,NPOLE2,DV,A,B,C,D,E,KKK,STEP)
      IF (KRE.EQ.3) GO TO 980
990   KRE=0
      ITER=ITER+1
      GO TO 370
1000  WRITE (6,1010)
1010  FORMAT ('0',5X,'*** TERMINATION REASON:  INITIAL COMPENSATORS DO N
1OT SATISFY ZETA CONSTRAINTS ***')
1020  STOP
C
      END

```

C

SUBROUTINE ADDPTS

```

SUBROUTINE ADDPTS (KPOINT,KIN,KOUT,NB,NM,STBM,KPTS,CT,G,GC,T,OMEGA
1,KGOBAK,KPTMAX,NIT,KINACT,NML,KACT,KPOLD,KOLD,KSYSM,C1,C1,WOPKI,ZA,
2ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N2,M1,M2,GAIN,KONT,A,B,C,D,E)

```

C

```

SUBPROGRAM DESIGNED TO GENERATE ADDITIONAL FREQUENCY DATA;
PROGRAM INCORPORATES THE ROUTINES CRT, INTER, AND TRFR.

```

C

SUBPROGRAM VARIABLES:

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

```

KPOINT - INTEGER NUMBER OF CURRENT DATA POINTS
KIN - INTEGER NUMBER OF CONTROLLER INPUTS
KOUT - INTEGER NUMBER OF CONTROLLER OUTPUTS
NB - STARTING NO. OF MARGINS TO BE INVESTIGATED
NM - INTEGER NUMBER OF MARGINS INVESTIGATED
STBM(I) - REAL ARRAY OF STABILITY MARGINS
KPTS(I) - INTEGER ARRAY OF FREQUENCY NOS. OF MARGINS
CT(I) - COMPLEX ARRAY OF FREQUENCY RESPONSE
G(I,J,K) - 3D COMPLEX ARRAY OF DISCRETE FREQ. RESPONSE
GC(I,J,K) - COMPLEX ARRAY OF COMPENSATION EVALUATED AT DISCRETE PT
T(I,J,K) - COMPLEX ARRAY OF TRANSFER RESPONSE Gc*G IS STORED
OMEGA(I) - COMPLEX ARRAY OF DISCRETE FREQUENCY POINTS
KPTMAX - MAXIMUM NUMBER OF FREQUENCY POINTS ALLOWABLE
NIT - AN INTEGER OF INACTIVE MARGINS
KINACT(I) - ARRAY OF INTEGERS CORRESPONDING TO INACTIVE MARGINS
NML - AN INTEGER DENOTING ACTIVE MARGINS DETECTED
KACT(I) - FREQUENCY DATA NUMBERS OF ACTIVE MARGINS
KPOLD - INTEGER OF DATA POINTS OF LAST ITERATION
KOLD(I) - INTEGER ARRAY OF PREVIOUS DATA POINTS
KSYSM - INTEGER REFERENCE TO PARTICULAR SUBSYSTEM

```

```

COMMON ITER,KSTART

```

```

INTEGER A,B,C,D,E

```

```

COMPLEX CT,G,GC,OMEGA,T

```

```

DIMENSION CT(D,A), G(B,A,D), GC(A,B,D), KACT(E,A), KINACT(E,A), KO
1LD(1), KPTS(E,A), NIT(A), NM(A), NML(A), OMEGA(D), STBM(1), T(A,A,
2D)

```

```

KGOBAK=0

```

```

NX=NM(KSYM)

```

```

DO 110 N=NB,NX

```

```

K=KPTS(N,KSYM)

```

```

DG1=CABS(CT(K,KSYM)-CT(K-1,KSYM))

```

```

DG2=CABS(CT(K+1,KSYM)-CT(K,KSYM))

```

```

IF (STBM(N).GT.5.*DG1) GO TO 100

```

```

K=K+1

```

```

IF (K.EQ.2) GO TO 100

```

10

```

CONTINUE

```

```

KPOINT=KPOINT+1

```

```

IF (KPOINT.GT.KPTMAX) RETURN

```

```

C   SEPARATING MATRICES TO ADD NEW FREQUENCY POINT
      DO 30 M=KPOINT,K,-1
      OMEGA(M)=OMEGA(M-1)
      DO 20 I=1,KIN
      CT(M,I)=CT(M-1,I)
      DO 20 J=1,KIN
20    T(I,J,M)=T(I,J,M-1)
      DO 30 I=1,KIN
      DO 30 J=1,KOUT
      G(J,I,M)=G(J,I,M-1)
30    GC(I,J,M)=GC(I,J,M-1)
      KGOBAK=1
      CALL INTER (OMEGA(K-1),OMEGA(K-2),OMEGA(K-1))
      DO 40 I=1,KOUT
      DO 40 J=1,KIN
40    CALL INTER (G(I,J,K-1),G(I,J,K-2),G(I,J,K-1))
      CALL EVAL (OMEGA(K-1),GC,G,T,K-1,KIN,KOUT,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,
1PD,PE,N1,M2,M1,M2,GAIN,KONT,A,B,C,D)
      DO 50 I=1,KSYM
      CALL CRT (1,K-1,I,T(1,1,K-1),KIN,KOUT,CT(1,I),G,PCG,I1,J1,C1,C1,WO
1RKI,A,B,D)
      NX=NM(I)
      DO 50 L=1,NX
50    IF (KPTS(L,I).GE.K-1) KPTS(L,I)=KPTS(L,I)+1
      IF (ITER.EQ.KSTART) GO TO 80
      DO 70 I=1,KIN
      NY=NIT(I)
      NZ=NML(I)
      DO 60 L=1,NY
60    IF (KINACT(L,I).GE.K-1) KINACT(L,I)=KINACT(L,I)+1
      DO 70 L=1,NZ
70    IF (KACT(L,I).GE.K-1) KACT(L,I)=KACT(L,I)+1
80    CONTINUE
      DO 90 L=1,KPOLD
90    IF (KOLD(L).GE.K-1) KOLD(L)=KOLD(L)+1
100   IF (STBM(N).GT.5.*DG2) GO TO 110
      STBM(N)=6.*DG2
      K=K+2
      IF (K.GT.KPOINT) GO TO 110
      GO TO 10
110  CONTINUE
      RETURN
C
      END

```

```

C                                     SUBROUTINE CHANGE
C
C      SUBROUTINE CHANGE(ZA,ZB,ZC,ZD,ZE,N1,N2,DV,DEL,KKK,KIN,KOUT,A,B,C)
C      DESIGNED TO CHANGE COMPENSATOR COEFFICIENTS ACCORDING TO THE DIRECT-
C      IONAL VECTOR
C
C      INTEGER A,B,C
C      DIMENSION ZA(A,B,C),ZB(A,B,C),ZC(A,B,C),ZD(A,B,C),ZE(A,B,C),DV(1)
C      ,N1(A,B),N2(A,B)
C
C      DO 4 I=1,KIN
C      DO 4 J=1,KOUT
C      NX=N1(I,J)
C      IF(NX.EQ.0)GO TO 2
C      DO 1 L=1,NX
C      ZA(I,J,L)=ZA(I,J,L)+DV(KKK)*DEL
C      ZB(I,J,L)=ZB(I,J,L)+DV(KKK+1)*DEL
C      KKK=KKK+2
C      NX=N2(I,J)
C      IF(NX.EQ.0) GO TO 4
C      DO 3 L=1,NX
C      ZC(I,J,L)=ZC(I,J,L)+DV(KKK)*DEL
C      ZD(I,J,L)=ZD(I,J,L)+DV(KKK+1)*DEL
C      ZE(I,J,L)=ZE(I,J,L)+DV(KKK+2)*DEL
C      KKK=KKK+3
C      CONTINUE
C      RETURN
C
C      END
C
C                                     SUBROUTINE CRT
C
C      SUBROUTINE CRT (KEY,K,KSTM,T,KIN,KOUT,CT,P,PCG,I1,J1,C1,C1,WORK1,A
C      1,B,D)
C
C      SUBPROGRAM PERFORMS 2 FUNCTIONS DENOTED BY THE INPUT KEY:
C      KEY IS 1, CLOSED LOOP FREQUENCY RESPONSE CT(S) IS FOUND IN
C      TERMS OF INPUT VECTOR; KEY IS 0, SUBPROGRAM AIDS PARTIAL IN
C      DETERMINING THE PARTIAL OF C(S) WRT. G(S)IJ. PROGRAM IN-
C      CORPORATES MATINC, MATMUL ROUTINES.
C
C      SUBPROGRAM VARIABLES:
C      KEY      - PROGRAM MODE VARIABLE
C      KSTM     - SUBSYSTEM IN CONSIDERATION
C      T(I,J)   - COMPLEX TRANSFER RESPONSE G(S)*P(S)
C      KIN      - NO. OF CONTROLLER INPUTS
C      KOUT     - NO. OF CONTROL OUTPUTS
C      P(I,J)   - COMPLEX PLANT RESPONSE ARRAY
C      I1       - CONTROL INPUT INDEX OF G(S)IJ
C      J1       - CONTROL OUTPUT INDEX OF G(S)IJ
C      CT(J)    - COMPLEX CLOSED LOOP FREQUENCY RESPONSE

```

```

C
C      TK IS THE KTH COLUMN OF T(S)
C      CI IS THE KSTM RESPONSE
C      ZEROING THE KTH COLUMN OF (T(S) + I):
C
      INTEGER A,B,D
      COMPLEX C1(A,A),T(A,A),CI(A,A),P(B,A),CT(D),PCG
      DO 10 I=1,KIN
      DO 10 J=1,KIN
      C1(I,J)=CMPLX(0.,0.)
      IF (I.NE.KSTM) C1(I,J)=T(I,J)
      IF (I.EQ.KSTM) C1(I,J)=CMPLX(0.,0.)
      IF (I.EQ.J) C1(I,J)=CMPLX(1.,0.)+C1(I,J)
10    CONTINUE
      CALL MATINC (C1,KIN,CI,IER,WORKI,A,A,2*A)
      IF (IER.EQ.2) GO TO 40
      CALL MATMUL (T,CI,C1,CI,T,T,KIN,KIN,KIN,KIN,1,0,A,A,1)
      CT(K)=C1(KSTM,KSTM)
C    C1 AT THIS POINT IS TOTAL RESPONSE C IN NOTES.
      IF (KEY.EQ.1) RETURN
C    NOW THE PARTIAL OF C W.R.T. G(I,J)
      PCG=0.0
      DO 20 I=1,KIN
20    PCG=PCG+P(J1,I)*CI(I,KSTM)
      IF (I1.EQ.KSTM) GO TO 30
      PCG=-C1(KSTM,I1)*PCG
30    CONTINUE
      RETURN
40    WRITE (6,50) KSTM,K
50    FORMAT (/5X,'INVERSE MATRIX INDETERMINATE BY THIS METHOD AT KSYM=
1',I3,'K= ',I3/)
      RETURN
C
      END

C
C      SUBROUTINE DELETE

      SUBROUTINE DELETE(KPOINT,KIN,KOUT,OMEGA,G,NIT,KINACT,NML,KACT,
CITER,KPOLO,KOLD,KNEW,A,B,C,D,E)

      INTEGER A,B,C,D,E
      COMPLEX OMEGA,G
      DIMENSION OMEGA(D),G(B,A,D),KINACT(E,A),NIT(A),KACT(E,A),KOLD(D),
C      CNML(A),KNEW(D)

      L=0
      DO 7 K=1,KPOINT
      DO 4 I=1,KIN
      NX=NIT(I)
      IF(NX.EQ.0) GO TO 2
      DO 1 J=1,NX
1    IF(K.EQ.KINACT(J,I)) GO TO 6
2    NX=NML(I)
      IF(NX.EQ.0) GO TO 4
4
6
7

```



```

DO 3 J=1,NX
3   IF(K.EQ.KACT(J,I)) GO TO 6
4   CONTINUE
DO 5 J=1,KPOLD
5   IF(K.EQ.KOLD(J)) GO TO 6
GO TO 7
6,  L=L+1
   KNEW(L)=K
7   CONTINUE
   KDELET=KPOINT-L
   KPOINT=L
DO 8 K=1,KPOINT
   L=KNEW(K)
   OMEGA(K)=OMEGA(L)
DO 8 I=1,KOUT
DO 8 J=1,KIN
8   G(I,J,K)=G(I,J,L)
DO 12 K=1,KPOINT
DO 12 I=1,KIN
NX=NIT(I)
DO 9 J=1,NX
9   IF(KNEW(K).EQ.KINACT(J,I))KINACT(J,I)=K
   NX=NML(I)
DO 10 J=1,NX
10  IF(KNEW(K).EQ.KACT(J,I))KACT(J,I)=K
DO 11 J=1,KPOLD
11  IF(KOLD(J).EQ.KNEW(K))KOLD(J)=K
12  CONTINUE
WRITE(6,13)KDELET,ITER
13  FORMAT(1H0,5X,I3,2X,"POINTS WERE DELETED ON ITERATION NO. "I3)
RETURN
C
END

```

```

C
C                                     SUBROUTINE DIRVEC

SUBROUTINE DIRVEC(G,N,KPARC,DV,WEIGHT,A1,AI,WORKR,E,F,Z,H)

C  DIRECTIONAL VECTOR PROGRAM
C
C  SUBPROGRAM DESIGNED TO CALCULATE THE DIRECTIONAL VECTOR OF
C  THE CONSTRAINT IMPROVEMENT ALGORITHM. DIRECTIONAL VECTOR
C  DV CALCULATED AS
C
C       $DV = (G) * A$ 
C  WHERE #G IS (N,M) GRADIENT MATRIX WHOSE COLUMNS ARE THE
C  GRADIENTS OF THE ACTIVE CONSTRAINTS; COLUMN VECTOR A IS
C       $A = INV(\#G * \#G) * C;$ 
C  COLUMN C IS M COMPONENT VECTOR OF POSITIVE ELEMENTS; M IS
C  NUMBER OF COMPENSATOR COEFFICIENTS.

```

```

C
C DEFINITIONS OF I/O VARIABLES
C
C G      -MATRIX WHOSE ROWS CONTAIN THE GRADIENT VECTORS OF THOSE
C          STABILITY MARGINS ONLY CONSIDERED PERTINENT
C NM     -NUMBER OF STABILITY MARGINS CONSIDERED PERTINENT
C KPARC  -NO. ROWS IN G, I.E., COMPENSATOR COEFFICIENTS
C DV(I)  -REAL ARRAY OF DIRECTIONAL VECTOR
C WEIGHT-WEIGHTING FACTOR VECTOR
C
      INTEGER E,F,Z,H
      DIMENSION G(E,F),A1(E,E),AI(E,E),WEIGHT(1),DV(1)
      DO 2 K=1,NM
      DO 2 J=K,NM
      SUM=0.
      DO 1 I=1,KPARC
1      SUM=SUM+G(J,I)*G(K,I)
      A1(J,K)= SUM
      A1(K,J)= SUM
2      CONTINUE
      IF(NM.GT.1) GO TO 3
      A1(1,1)=1./A1(1,1)
      A1(1,1)= WEIGHT(1) * A1(1,1)
      GO TO 6
3      CONTINUE
      CALL MATINV(A1,NM,AI,IER,WORKR,E,E,H)
      IF(IER.EQ.0) GO TO 5
      WRITE(6,4)
4      FORMAT('0',15X,'**** TERMINATION: PARTIALS NOT LINEARLY INDEPENDENT
      CNT ****')
      STOP
5      CALL MATMUL(AI,Y,Y,AI,WEIGHT,A1,NM,NM,NM,1,1,1,E,E,1)
6      CONTINUE
      DO 8 I=1,KPARC
      SUM= 0.
      DO 7 J=1,NM
7      SUM=SUM+G(J,I)*A1(J,1)
8      DV(I)=SUM
      RETURN
      END

C
C SUBROUTINE EVAL

      SUBROUTINE EVAL (XF,GC,G,T,K,KIN,KOUT,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,P
      1E,N1,N2,M1,M2,GAIN,KONT,A,B,C,D)

      INTEGER A,B,C,D
      COMPLEX GCX,GCY,GCN,GCD,GC(A,B,D),G(B,A,D),T(A,A,D)
      DIMENSION N1(A,B), COEF(3), ZA(A,B,C), ZB(A,B,C), ZC(A,B,C), ZD(A,
      1B,C), ZE(A,B,C), PA(A,B,C), PB(A,B,C), PC(A,B,C), PD(A,B,C), PE(A,
      2B,C), GAIN(A,B), N2(A,B), M1(A,B), M2(A,B)
      COMPLEX XF

```

```

C      EVALUATION OF GC, THE COMPENSATOR TRANSFER
DO 90 I=1,KIN
DO 90 J=1,KOUT
GCX=CMPLX(1.,0.)
GCY=CMPLX(1.,0)
GCN=CMPLX(1.,0.)
GCD=CMPLX(1.,0.)
NC=N1(I,J)
IF (NC.EQ.0) GO TO 20
DO 10 L=1,NC
COEF(1)=ZA(I,J,L)
COEF(2)=ZB(I,J,L)
CALL POLEV (COEF,1,XF,GCX)
GCN=GCX*GCN
10  CONTINUE
20  MC=M1(I,J)
IF (MC.EQ.0) GO TO 40
DO 30 L=1,MC
COEF(1)=PA(I,J,L)
COEF(2)=PB(I,J,L)
CALL POLEV (COEF,1,XF,GCY)
GCD=GCY*GCD
30  CONTINUE
40  IF (GCD.EQ.0.) GCD=CMPLX(1.,0.)
GCN=GCN/GCD
NC=N2(I,J)
IF (NC.EQ.0) GO TO 60
DO 50 L=1,NC
COEF(1)=ZC(I,J,L)
COEF(2)=ZD(I,J,L)
COEF(3)=ZE(I,J,L)
CALL POLEV (COEF,2,XF,GCX)
GCN=GCX*GCN
50  CONTINUE
60  MC=M2(I,J)
GCD=CMPLX(1.,0.)
IF (MC.EQ.0) GO TO 80
DO 70 L=1,MC
COEF(1)=PC(I,J,L)
COEF(2)=PD(I,J,L)
COEF(3)=PE(I,J,L)
CALL POLEV (COEF,2,XF,GCY)
GCD=GCY*GCD
70  CONTINUE
80  CONTINUE
IF (GCD.EQ.0.) GCD=CMPLX(1.,0.)
GC(I,J,K)=GAIN(I,J)*GCN/GCD
90  CONTINUE
C      GC*G TRANSFER FUNCTION
C      HERE STARTS TRANSFER GC*G O.L. FREQUENCY RESPONSE
CALL MATMUL (GC(1,1,K),G(1,1,K),T,GC,GC,GC,KIN,KOUT,KOUT,KIN,K,0,A
1,B,B)
RETURN
C
END

```

```

C                               SUBROUTINE GAINMG

      SUBROUTINE GAINMG (GTOTAL,KPOINT,NM,FQMIN,FQMAX,KPTS,STBM,OMEGA)
C SUBPROGRAM FOR CALCULATING GAIN MARGINS

C DEFINITIONS OF I/O VARIABLES
C
C GTOTAL-COMPLEX ARRAY OF COMPENSATED OPEN FREQ. RESPONSE
C KPOINT-NO. OF POINTS
C OMEGA -ARRAY OF FREQS.
C NM      -COUNTER
C KPTS    -FREQUENCY NOS. WHERE MARGINS OCCUR
C STBM    -STABILITY MARGINS OF MARGINS
C FQMIN   -LOWER FREQ. FOR MARGIN DETECTION
C FQMAX   -UPPER FREQ. FOR MARGIN DETECTION

      COMPLEX OMEGA,GTOTAL
      DIMENSION GTOTAL(1), KPTS(1), STBM(1), OMEGA(1)

      P=1.0
      DO 30 I=1,KPOINT
      S2=AIMAG(GTOTAL(I))
      IF (I.EQ.1) S1=S2
      IF (AIMAG(OMEGA(I)).GT.FQMAX) RETURN
      IF (AIMAG(OMEGA(I)).LT.FQMIN) GO TO 20
      IF (ABS(S2).LT.1.0E-20) GO TO 10
      SGN=S2/ABS(S2)
      IF (S1*SGN.GT.0.0) GO TO 20
10  IF ((REAL(GTOTAL(I)).GE.0.).AND.(REAL(GTOTAL(I-1)).GE.0.)) GO TO 2
      10
      I1=I-1
      IF (ABS(S2).LT.ABS(S1)) I1=I
      NM=NM+1
      KPTS(NM)=I1
      FRAC=S1/(S1-S2)
      STBM(NM)=CABS(P+FRAC*GTOTAL(I)+(1.-FRAC)*GTOTAL(I-1))
20  S1=S2
30  CONTINUE
      RETURN
C
      END

```

```

C                               SUBROUTINE INTER

SUBROUTINE INTER(S,T,R)
C
C   INTERPOLATION SUBPROGRAM
C
C   SUBPROGRAM INTERPOLATES SPECIFIED INPUT DATA IN
C   CONJUNCTION WITH ADDPTS ROUTINE; MAGNITUDES ARE
C   INTERPOLATED LOGARITHMICALLY, PHASES LINEARLY.
C
C   SUBPROGRAM VARIABLES:
C   S - COMPLEX LOWER BOUND OF QUANTITY INTERPOLATED
C   T - COMPLEX UPPER BOUND OF QUANTITY INTERPOLATED
C   R - COMPLEX RESULTANT OF THE INTERPOLATION
C
C   COMPLEX S,T,R
C   DATA PI/3.14159265358979/
C
C   X=SQRT(CABS(S)*CABS(T))

Y=0.
  IF((X.GT.1.E-10)) GO TO 1
  GO TO 2
1  CONTINUE
  U=ATAN2(AIMAG(S),REAL(S))
  V=ATAN2(AIMAG(T),REAL(T))
  IF(U.LT.0.) V=V+2.*PI
  IF(V.LT.0.) V=V+2.*PI
  Y=(U+V)/2.
  IF(ABS(U-V).GT.PI)Y=Y-PI
  IF(ABS(Y-U).GT.PI/2.)Y=Y-PI
2  R=CMPLX(X*COS(Y),X*SIN(Y))
  RETURN
C
  END

```

```

C                               SUBROUTINE MATINC

SUBROUTINE MATINC (XX,N,YY,IER,X,A,B,C)

C   SUBROUTINE FOR FINDING AN INVERSE OF A MATRIX
C
C   X = MATRIX FOR WHICH INVERSE IS TO BE TAKEN
C   N = DIMENSION OF SQUARE MATRIX X
C   Y = INVERSE OF X
C   IER = ERROR CODE
C       IER= 0 - NO ERROR EXISTS
C       IER= 2 - MATRIX DOES NOT POSSESS AN INVERSE
C
C   INTEGER A,B,C
C   DIMENSION XX(A,A), YY(A,A), X(A,C)
C   IMPLICIT COMPLEX (D-H,O-Z)
C
C
C   DO 10 I=1,N
C   DO 10 J=1,N
C   X(I,J)=XX(I,J)
10  CONTINUE
C   IER=0
C   N2=N+N
C   N1=N+1
C   DO 20 I=1,N
C   DO 20 K=N1,N2
C   L=K-N
C   IF (L.NE.1) X(I,K)=CMPLX(0.,0.)
C   IF (L.EQ.1) X(I,K)=CMPLX(1.,0.)
20  CONTINUE
C   DO 50 I=1,N
C   IF (CABS(X(I,I)).LT.1.0E-25) GO TO 70
C   SCRW=X(I,I)
C   DO 30 K=1,N2
C   X(I,K)=X(I,K)/SCRW
30  DO 50 L=1,N
C   IF (L.EQ.1) GO TO 50
C   SLOP=X(L,I)
C   DO 40 K=1,N2
C   X(L,K)=X(L,K)-SLOP*X(I,K)
40  CONTINUE
50  CONTINUE
C   DO 60 I=1,N
C   DO 60 K=N1,N2
C   L=K-N
C   YY(I,L)=X(I,K)
60  GO TO 80
70  IER=2
80  RETURN
C
END

```

```

C                               SUBROUTINE MATINV

SUBROUTINE MATINV (XX,N,YY,IER,X,E,Z,H)

INTEGER E,Z,H
DIMENSION XX(E,Z), YY(E,Z), X(E,H)

DO 10 I=1,N
DO 10 J=1,N
10 X(I,J)=XX(I,J)
N2=N+N
N1=N+1
DO 20 I=1,N
DO 20 K=N1,N2
L=K-N
IF (L.NE.I) X(I,K)=0.
IF (L.EQ.I) X(I,K)=1.
20 CONTINUE
DO 50 I=1,N
IF (ABS(X(I,I)).LT.1.E-25) GO TO 70
SCRW=X(I,I)
DO 30 K=1,N2
30 X(I,K)=X(I,K)/SCRW
DO 50 L=1,N
IF (L.EQ.I) GO TO 50
SLOP=X(L,I)
DO 40 K=1,N2
X(L,K)=X(L,K)-SLOP*X(I,K)
40 CONTINUE
50 CONTINUE
DO 60 I=1,N
DO 60 K=N1,N2
L=K-N
60 YY(I,L)=X(I,K)
GO TO 80
70 IER=2
80 RETURN
C
END

C                               SUBROUTINE MATMUL

SUBROUTINE MATMUL (AC,BC,CC,AR,BR,CR,N,L,L1,M,ND,NC,A,B,C)

C          SUBROUTINE MATRIX MULTIPLICATION
C
C          MATRIX MULTIPLICATION MULTIPLIES A * B
C          MATRIX AC IS N X L
C          MATRIX BC IS L X M
C          MATRIX CC IS THE RESULTANT MATRIX N X M
C          MATRIX CR IS REAL RESULTANT
C          NC IS COMPLEX KEY:
C          NC=0; AC,BC,CC, ARE COMPLEX MATRICES
C          NC=1F AR,BR,CR,ARE REAL MATRICES
C

```

```

C      INTEGER A,B,C
      COMPLEX AC,BC,CC
      DIMENSION AC(A,B), BC(B,A), CC(A,A,ND), AR(A,B), BR(B,C), CR(A,C,N
1D)
C      IF (NC.EG.1) GO TO 20
      DO 10 I=1,N
      DO 10 J=1,M
      CC(I,J,ND)=CMPLX(0.,0.)
      DO 10 K=1,L
10     CC(I,J,ND)=CC(I,J,ND)+AC(I,K)*BC(K,J)
      RETURN
20     DO 30 I=1,N
      DO 30 J=1,M
      CR(I,J,ND)=0.
      DO 30 K=1,L
30     CR(I,J,ND)=CR(I,J,ND)+AR(I,K)*BR(K,J)
      RETURN
C
      END

```

SUBROUTINE NYQIST

```

C      SUBROUTINE NYQIST (N,G,NRHP,NCON,NCIRL,NZ,FMAX,F)

      COMPLEX G(1),F(1)
      DOUBLE PRECISION PI
      DATA PI /3.14159265358979D+00/

      KC=NCON/2
      LC=(NCON+1)/2
      C1=ATAN2(AIMAG(G(1)),1.0+REAL(G(1)))
      SUM=-C1+ABS(ATAN2(0.0,1.0+REAL(G(1))))*ABS(C1)/C1
      IF (KC.NE.LC) SUM=0.0
      DO 10 I=2,N
      C2=ATAN2(AIMAG(G(I)),1.0+REAL(G(I)))
      DIFF=C2-C1
      IF (ABS(DIFF).GT.PI) DIFF=C2-C1+2.0*PI*ABS(C1)/C1
      SUM=SUM-DIFF
      IF (AIMAG(F(I)).GT.FMAX) GO TO 20
10     C1=C2
      I=N
20     SUM=+C2-ABS(ATAN2(0.0,1.0+REAL(G(I))))*ABS(C2)/C2+SUM
      SUM=2.0*(SUM+C2)+PI*NCON
      SUM=SUM+ABS(SUM)*PI/(4.0*SUM)
      NCIRL=SUM/(2.0*PI)
      NZ=NRHP+NCIRL
      RETURN
C
      END

```


SUBROUTINE OUTPT

SUBROUTINE OUTPT (N, KSYM, KPOINT, CT, OMEGA, ITER, NM, KPTS, KMIN, STBM, RQ
1, TYPE, ACTIVE, KIN, KOUT, ZA, ZB, ZC, ZD, ZE, PA, PB, PC, PD, PE, N1, N2, M1, M2, GA
2IN, KONT, A, B, C, KOLD)

SUBPROGRAM DESIGNED TO OUTPUT INFORMATION IN THREE AREAS
AS DESIGNATED BY THE KEY N:

KEY:

N=0, OUTPUT COMPENSATOR INFORMATION
N=1, OUTPUT FREQUENCY RESPONSE
N=2, OUTPUT STABILITY MARGINS INFORMATION
N=3, COMPLETE OUTPUT INFORMATION

INTEGER A, B, C

INTEGER TYPE, ACTIVE, XXP, ZDUMB

COMPLEX CT(1), OMEGA(1), X

DIMENSION GAIN(A, B), KONT(A, B), N1(A, B), N2(A, B), M1(A, B), M2(A, B)
1, ZA(A, B, C), ZB(A, B, C), ZC(A, B, C), ZD(A, B, C), ZE(A, B, C), PA(A, B, C)
2, PB(A, B, C), PC(A, B, C), PD(A, B, C), PE(A, B, C), KPTS(1), STBM(1), RQ
3(1), ACTIVE(1), TYPE(1), KOLD(1)

DATA IBK, IAT, XXP, RAD2 /1H ,1H*,1HP,114.591559/

IF (N.EQ.2) GO TO 170

IF (N.EQ.1) GO TO 110

IF (KSYM.EQ.1) WRITE (6,20)

DO 100 I=1,KIN

DO 100 J=1,KOUT

IF (GAIN(I,J).EQ.0.) GO TO 80

WRITE (6,30) I,J,GAIN(I,J)

NC=N1(I,J)

IF (NC.NE.0) WRITE (6,40) (ZA(I,J,L),ZB(I,J,L),L=1,NC)

NC=N2(I,J)

IF (NC.NE.0) WRITE (6,50) (ZC(I,J,L),ZD(I,J,L),ZE(I,J,L),L=1,NC)

MC=M1(I,J)

IF (MC.NE.0) WRITE (6,60) (PA(I,J,L),PB(I,J,L),L=1,MC)

MC=M2(I,J)

IF (MC.NE.0) WRITE (6,70) (PC(I,J,L),PD(I,J,L),PE(I,J,L),L=1,MC)

WRITE (6,10) I,J,KONT(I,J)

10 FORMAT ('0',5X,' DC GAIN CONSTRAINT FOR EACH CHANNEL (IF KONT=1,
1 ALLOWED TO VARY; IF KONT =2 , HELD CONSTANT)',/,7X,'KONT(',13,' ,
2',13,') = ',13,/,)

20 FORMAT (/,5X,'THE COMPENSATION ELEMENTS ARE DESCRIBED BY TRANSFER
1 FUNCTIONS IN',/,5X,'CASCADED FIRST AND SECOND ORDER FACTORS:',/,3
21X,'N1',14X,'N2',/,31X,'PR (ZA + ZB S) PR (ZC + ZD S + ZE S**2)',
3/,30X,'I=1',4X,'I',5X,'I',3X,'J=1',3X,'J'5X,'J'6X,'J',/,18X,'(GAIN)
4-----',/,31X,'M
51',14X,'M2',/,31X,'PR (PA + PB S) PR (PC + PD S + PE S**2)',/,30
6X,'I=1 I I J=1 J J J')/)

30 FORMAT (/,5X,'COMPENSATOR(',13,',',13,'): GAIN = ',G10,5,/,)

40 FORMAT (15X,'COMPENSATOR COEFFICIENTS:',/,15X,'ZA = ',G12,6,10X,')

```

128 = ',G12.6))
50  FORMAT (15X,'ZC = ',G12.6,10X,'ZD = ',G12.6,10X,'ZE = ',G12.6)
60  FORMAT (15X,'PA = ',G12.6,10X,'PB = ',G12.6)
70  FORMAT (15X,'PC = ',G12.6,10X,'PD = ',G12.6,10X,'PE = ',G12.6)
    GO TO 100
80  WRITE (6,90) I,J
90  FORMAT (10X,'COMPENSATOR(',I3,',',I3,') HAS ZERO CONTRIBUTION.')
100 CONTINUE
    IF (N.NE.3) RETURN
110 WRITE (6,120) KSYM
120 FORMAT (/,5X,'THE FREQUENCY RESPONSE OF SYSTEM NO. ',I3,' IS: ',/,
17X,'DATA',10X,'COMPLEX OMEGA',10X,'COMPLEX CT',10X,'MAG & PHASE',/
2)
    L=1
    DO 140 K=1,KPOINT
        ZDUMB=IBK
        IF (K.NE.KOLD(L)) GO TO 130
        ZDUMB=IAT
        L=L+1
130 CONTINUE
        X=CMPLX(CABS(CT(K)),ATAN2(AIMAG(CT(K)),REAL(CT(K)))*57.295779)
140 WRITE (6,160) ZDUMB,K,OMEGA(K),CT(K),X
        WRITE (6,150)
150 FORMAT ('0',T9,'* DENOTES ORIGINAL FREQUENCY POINTS')
160 FORMAT (8X,A1,I3,5X,2G10.4,5X,2G10.4,5X,2G10.4)
        IF (N.NE.3) RETURN
170 WRITE (6,180) KSYM,ITER
180 FORMAT ('0',25X,'SYSTEM NO. ',I3,', ITERATION NO. ',I4)
        DO 270 I=1,NM
            K=KPTS(I)
            IF (I.EQ.KMIN+1) GO TO 190
            IF (I.EQ.1) GO TO 220
            GO TO 240
190 WRITE (6,200)
200 FORMAT ('0',25X,'ATTENUATED FREQUENCY INFORMATION')
        WRITE (6,210)
210 FORMAT ('0',T2,'NO.',T7,'MARGIN RADIUS',T26,'FREQUENCY',T41,'DESIR
1ED MARGIN',T57,'MARGIN TYPE',T70,'ACTIVE')
        GO TO 240
220 WRITE (6,230)
        WRITE (6,210)
230 FORMAT ('0',25X,'RELATIVE STABILITY INFORMATION')
240 XDUMB=STBM(I)
        YDUMB=RQ(I)
        IF (TYPE(I).NE.XXP) GO TO 250
        XDUMB=RAD2*ASIN(XDUMB/2.)
        YDUMB=RAD2*ASIN(YDUMB/2.)
250 WRITE (6,260) I,XDUMB,OMEGA(K),YDUMB,TYPE(I),ACTIVE(I)
260 FORMAT (' ',T2,I2,T8,G10.4,T20,G10.4,T31,G10.4,T43,G10.4,T63,A1,T7
12,A3)
270 CONTINUE
    RETURN
C
    END

```

C

SUBROUTINE PARTAL

```

SUBROUTINE PARTAL (OMEGA,NFREQ,CT,KPTS,TYPE,T,P,G,KSYSM,C1,CI,WORKI
1,A,B,C,D,E,F,Z,PFY,KIN,KOUT,ZA,ZB,ZC,ZD,ZE,PA,PB,PC,PD,PE,N1,N
22,M1,M2,GAIN,KONT)

COMPLEX ANUM,PCG,G(A,B,D),OMEGA(D),Q,XV,CT(D),DIST,PGX(3),T(A,A,D)
1,P(B,A,D)
INTEGER A,B,C,D,E,F,Z
INTEGER TYPE,XXT
DIMENSION N1(A,B), N2(A,B), M1(A,B), M2(A,B), ZA(A,B,C), ZB(A,B,C)
1, ZC(A,B,C), ZD(A,B,C), ZE(A,B,C), PA(A,B,C), PB(A,B,C), PC(A,B,C)
2, PD(A,B,C), PE(A,B,C), KONT(A,B), GAIN(A,B)
DIMENSION KPTS(1), TYPE(1), XXT(4), PFY(E,Z), PFY(E,Z)
DATA IBLANK,XXT /4H      ,1HG,1HP,1HS,1HA/

C
NOP=0
DO 190 K=1,NFREQ
KWHICH=KPTS(K)
SGN=+1.
IF (TYPE(K).EQ.XXT(4)) SGN=-1.
IF (TYPE(K).EQ.XXT(1)) GO TO 10
IF (TYPE(K).EQ.XXT(2)) GO TO 30
IF (TYPE(K).EQ.XXT(3)) Q=CMPLX(-1.,0.)
IF (TYPE(K).EQ.XXT(4)) Q=CMPLX(0.,0.)
GO TO 60
10 DO 20 L=0,1
20 IF (AIMAG(CT(KWHICH+L))*AIMAG(CT(KWHICH+L-1)).LE.0.) GO TO 50
30 DO 40 L=0,1
40 IF ((CABS(CT(KWHICH+L))-1.)*(CABS(CT(KWHICH+L-1))-1.).LE.0.) GO TO
1 50
50 DIST=CT(KWHICH+L)-CT(KWHICH+L-1)
XV=CONJG(CT(KWHICH)+1.)
DIST=CONJG(DIST)/CABS(DIST)
DIST=CMPLX(AIMAG(DIST),REAL(DIST))
IF (REAL(DIST*XV).GT.0.) DIST=-DIST
Q=CT(KWHICH)+5.*DIST
60 DIST=CONJG(-Q+CT(KWHICH))
XV=OMEGA(KWHICH)
KNOT=0
LNOT=0
DO 180 I=1,KIN
DO 180 J=1,KOUT
IOP=KONT(I,J)
CALL CRT (2,KWHICH,KSYSM,T(1,1,KWHICH),KIN,KOUT,CT,P(1,1,KWHICH),PC
1G,I,J,C1,CI,WORKI,A,B,D)
PCG=PCG*GAIN(I,J)
NCOMD=N1(I,J)
IF (NCOMD.EQ.0) GO TO 90

```

```

      DO 80 N=1,NCOMD
      IF (N.GT.1) IOP=2
      ANUM=ZA(I,J,N)+ZB(I,J,N)*XV
      PGX(1)=G(I,J,KWHICH)/ANUM
      PGX(2)=PGX(1)*XV
      DO 70 L=1,2
70    PFX(K,KNOT+L)=2.*REAL(PCG*PGX(L)*DIST*SGN)
      IF (IOP.EQ.2) PFX(K,KNOT+1)=0.0
80    KNOT=KNOT+2
90    NCOMD=N2(I,J)
      IF (NCOMD.EQ.0) GO TO 120
      DO 110 N=1,NCOMD
      IF (N.GT.1) IOP=2
      ANUM=ZC(I,J,N)+(ZD(I,J,N)+ZE(I,J,N)*XV)*XV
      PGX(1)=G(I,J,KWHICH)/ANUM
      PGX(2)=PGX(1)*XV
      PGX(3)=PGX(2)*XV
      DO 100 L=1,3
100   PFX(K,KNOT+L)=2.*REAL(PCG*PGX(L)*DIST*SGN)
      IF (IOP.EQ.2) PFX(K,KNOT+1)=0.0
110   KNOT=KNOT+3
C    DENOMINATOR PARTIALS
120   NCOMD=M1(I,J)
      IF (NCOMD.EQ.0) GO TO 150
      DO 140 N=1,NCOMD
      IF (N.GT.1) IOP=2
      ANUM=PA(I,J,N)+PB(I,J,N)*XV
      PGX(1)=G(I,J,KWHICH)/ANUM
      PGX(2)=PGX(1)*XV
      IF (IOP.EQ.2) PGX(1)=CMPLX(0.,0.)
      DO 130 L=1,2
130   PFY(K,LNOT+L)=-2.*REAL(PCG*PGX(L)*DIST*SGN)
      IF (IOP.EQ.2) PFY(K,LNOT+1)=0.0
140   LNOT=LNOT+2
150   NCOMD=M2(I,J)
      IF (NCOMD.EQ.0) GO TO 180
      DO 170 N=1,NCOMD
      IF (N.GT.1) IOP=2
      ANUM=PC(I,J,N)+(PD(I,J,N)+PE(I,J,N)*XV)*XV
      PGX(1)=G(I,J,KWHICH)/ANUM
      PGX(2)=PGX(1)*XV
      PGX(3)=PGX(2)*XV
      IF (IOP.EQ.2) PGX(1)=CMPLX(0.,0.)
      DO 160 L=1,3
160   PFY(K,LNOT+L)=-2.*REAL(PCG*PGX(L)*DIST*SGN)
      IF (IOP.EQ.2) PFY(K,LNOT+1)=0.0
170   LNOT=LNOT+3
180   CONTINUE
190   CONTINUE
      RETURN
C
      END

```

```

C                                     SUBROUTINE PHASEM

      SUBROUTINE PHASEM(GTOTAL,KPOINT,NM,FQMIN,FQMAX,KPTS,STBM,OMEGA)

C SUBPROGRAM FOR CALCULATING PHASE MARGINS
C
C DEFINITIONS OF I/O VARIABLES
C
C GTOTAL-COMPLEX ARRAY OF COMPENSATED OPEN LOOP FREQ. RESPONSE
C KPOINT-NO. OF POINTS
C OMEGA -ARRAY OF FREQS.
C NM      -COUNTER
C KPTS    -FREQUENCY NOS. WHERE MARGINS OCCUR
C STBM    -STABILITY MARGINS OF MARGINS
C FQMIN   -LOWER FREQ. FOR MARGIN DETECTION
C FQMAX   -UPPER FREQ. FOR MARGIN DETECTION
C
      DIMENSION GTOTAL(1),KPTS(1),STBM(1),OMEGA(1)
      COMPLEX OMEGA,GTOTAL
C
      P=1.0
      DO 3 I=1,KPOINT
      S0=CABS(GTOTAL(I))
      S2=S0-1.0
      IF(I.EQ.1)S1=S2
      IF(AIMAG(OMEGA(I)).GT.FQMAX)RETURN
      IF(AIMAG(OMEGA(I)).LT.FQMIN)GO TO 2
      IF(ABS(S2).LT.1.0E-20)GO TO 1
      SGN=S2/ABS(S2)
      IF(S1*SGN.GT.0.0)GO TO 2
1      I1=I-1
      IF(ABS(S2).LT.ABS(S1))I1=I
      NM=NM+1
      KPTS(NM)=I1
      IF(S1.EQ.S2)S1=1.E-09
      FRAC=S1/(S1-S2)
      STBM(NM)=CABS(P+FRAC*GTOTAL(I)+(1.-FRAC)*GTOTAL(I-1))
2      S1=S2
3      CONTINUE
      RETURN
      END
C
C
C
C
C                                     SUBROUTINE POLEV
C
      SUBROUTINE POLEV(FW,K,X,F)

C PROGRAM FOR EVALUATING A POLYNOMIAL AT A COMPLEX FREQUENCY
C
C DEFINITIONS OF I/O VARIABLES
C
C FW      -VECTOR POLYNOMIAL COEFFICIENTS
C K        -ORDER OF POLYNOMIAL
C X        -COMPLEX FREQUENCY OF EVALUATION

```

```

C      COMPLEX X,SUM,F
C      DIMENSION FW(1)
C
C      SUM=FW(K+1)
C      IF(K.EQ.0) GO TO 20
C      DO 10 I=K,1,-1
10     SUM=FW(I) + SUM*X
20     F=SUM
C      RETURN
C
C      END

C
C      SUBROUTINE SRMINS

C      SUBROUTINE SRMINS (GTOTAL,KPOINT,NM,P,N,FQMIN,FQMAX,KPTS,STBM,OMEGA
C      1A)
C
C      SUBPROGRAM FOR DETERMINING THE MINMUNS OF THE OPEN LOOP FREQUENCY
C      RESPONSE WITH RESPECT TO THE -1 POINT WHEN GIVEN POINTS ON THE
C      OPEN LOOP FREQUENCY RESPONSE
C
C      DESCRIPTION OF I/O VARIABLES
C      KPOINT - NUMBER POINTS OF THE OPEN LOOP FREQ. RESPONSE GIVEN
C      OMEGA - CORRESPONDING FREQUENCIES OF CHOSEN POINTS
C      KPTS - FREQUENCY NOS. WHERE MARGINS OCCUR
C      FQMIN - MINIMUM FRQ. CONSIDERED
C      -P - POINT W.R.T. A MAX. OR MIN. IS DESIRED
C      N - DETERMINES WHETHER A MAX. OR MIN. IS DETERMINED
C
C      COMPLEX OMEGA,GTOTAL
C      DIMENSION GTOTAL(1), KPTS(1), STBM(1), OMEGA(1)
C
C      ASN1=0.0
C      S1=0.0
C      IF (N.LT.0) S1=1.0E15
C      DO 50 I=1,KPOINT
C      IF (AIMAG(OMEGA(I)).GT.FQMAX) RETURN
C      IF (AIMAG(OMEGA(I)).LT.FQMIN) GO TO 50
C      S2=CABS(P+GTOTAL(I))**2
C      ASN2=S2-S1
C      IF (ASN2*4) 10,50,10
10     IF (ASN1*ASN2) 20,40,40
20     IF (ASN1*N) 30,40,40
30     NM=NM+1
C      I1=I-1
C      KPTS(NM)=I1
C      STBM(NM)=SQRT(S1)
40     S1=S2
C      ASN1=ASN2
50     CONTINUE
C      RETURN
C
C      END

```

C

SUBROUTINE XCHECK

C

```

SUBROUTINE XCHECK (KIN,KOUT,ZA,ZB,ZC,ZD,ZE,N1,N2,DV,ZETA,PG,NAM,K
1E,NRTR1,NRTR2,LPV,A,B,C,D,E,F,KKK)

```

C

```

INTEGER A,B,C,D,E,F
DIMENSION ZA(A,B,C), ZB(A,B,C), ZC(A,B,C), ZD(A,B,C), ZE(A,B,C),
11(A,B), N2(A,B), PG(E,F), DV(1), KJUMP(3)

```

10

20

30

40

50

40

```

DO 100 I=1,KIN
DO 100 J=1,KOUT
K1=N1(I,J)
IF (K1.EQ.0) GO TO 50
DO 40 K=1,K1
IF (NRTR1.EQ.1) GO TO 40
IF (ZA(I,J,K).GT.1.E-05) GO TO 20
IF (ZA(I,J,K).LT.0.) GO TO 110
IF (DV(KKK+1).GE.0.) GO TO 20
KRE=1
DO 10 L=1,NAM
PG(L,KKK+1)=0.
LPV=LPV-1
CONTINUE
IF (ZB(I,J,K).GT.1.E-05) GO TO 40
IF (ZB(I,J,K).LT.0.) GO TO 110
IF (DV(KKK+2).GE.0.) GO TO 40
KRE=1
DO 30 L=1,NAM
PG(L,KKK+2)=0.
LPV=LPV-1
KKK=KKK+2
K2=N2(I,J)
IF (K2.EQ.0) GO TO 100
DO 90 K=1,K2
IF (NRTR2.EQ.1) GO TO 90
WN=SQRT(ZC(I,J,K)/ZE(I,J,K))
ZT=ZD(I,J,K)/(2.*WN*ZE(I,J,K))
IF (ZT.GE.ZETA+1.E-03) GO TO 90
IF (ZT.LT.ZETA) GO TO 110
PWN=(ZE(I,J,K)*DV(KKK+1)-ZC(I,J,K)*DV(KKK+3))/(2.*WN*ZE(I,J,K)**2)
AINC=-STEP/PMG
DEL=-AINC
WN=SQRT(ZC(I,J,K)/ZE(I,J,K))
ZET=ZD(I,J,K)/(2.*WN*ZE(I,J,K))
IF (ZET.GE.ZETA+1.E-03) GO TO 50
IF (ZET.LT.ZETA) GO TO 40
GO TO 50
DEL=DEL/2.
AINC=AINC+DEL*SGN

```

```

A0=ZC(I,J,K)+AINC*DV(K2+1)
A1=ZD(I,J,K)+AINC*DV(K2+2)
A2=ZE(I,J,K)+AINC*DV(K2+3)
ZET=A1/(2.*A2*SQRT(A0/A2))
IF (ZET.LT.ZETA) SGN=-1.0
IF (ZET.GE.ZETA+0.999E-03) SGN=1.0.
IF (ZET.LT.ZETA) GO TO 40
IF (ZET.GE.ZETA+0.999E-03) GO TO 40
CALL SELECT (STEP,STPNEW,AINC*PMG)
KRE=3
50 K2=K2+3
60 CONTINUE
RETURN
C
SUBROUTINE SELECT (STEP,STPNEW,STPTRY)
IF (STEP.GE.0.) STPNEW=AMIN1(STPNEW,STPTRY)
IF (STEP.LT.0.) STPNEW=AMAX1(STPNEW,STPTRY)
RETURN
C
END
C
SUBROUTINE YCHECK

SUBROUTINE YCHECK (KIN,KOUT,N1,N2,ZA,ZB,ZC,ZD,ZE,KRE,STEP,PMG,ZETA
1,NRTR1,NRTR2,DV,A,B,C,D,E,K2,STPNEW)

INTEGER A,B,C,D,E
DIMENSION ZA(A,B,C), ZB(A,B,C), ZC(A,B,C), ZD(A,B,C), ZE(A,B,C), N
11(A,B), N2(A,B), DV(1)
C
DO 60 I=1,KIN
DO 60 J=1,KOUT
K1=N1(I,J)
IF (K1.EQ.0) GO TO 30
DO 20 K=1,K1
IF (NRTR1.EQ.1) GO TO 20
IF (ZA(I,J,K).GE.0.) GO TO 10
KRE=3
DEL=ZA(I,J,K)/DV(K2+1)
CALL SELECT (STEP,STPNEW,(DEL+.000001)*PMG)
10 IF (ZB(I,J,K).GE.0.) GO TO 20
KRE=3
DEL=ZB(I,J,K)/DV(K2+2)
CALL SELECT (STEP,STPNEW,(DEL+.000001)*PMG)
20 K2=K2+2
30 CONTINUE
K1=N2(I,J)
IF (K1.EQ.0) GO TO 60
DO 50 K=1,K1
IF (NRTR2.EQ.1) GO TO 50
SGN=1.0
AINC=-STEP/PMG
DEL=-AINC
WN=SQRT(ZC(I,J,K)/ZE(I,J,K))
ZET=ZD(I,J,K)/(2.*WN*ZE(I,J,K))

```



```

      IF (ZET.GE.ZETA+1.E-03) GO TO 50
      IF (ZET.LT.ZETA) GO TO 40
      GO TO 50
40    DEL=DEL/2.
      AINC=AINC+DEL*SGN
      A0=ZC(I,J,K)+AINC*DV(K2+1)
      A1=ZD(I,J,K)+AINC*DV(K2+2)
      A2=ZE(I,J,K)+AINC*DV(K2+3)
      ZET=A1/(2.*A2*SQRT(A0/A2))
      IF (ZET.LT.ZETA) SGN=-1.0
      IF (ZET.GE.ZETA+0.999E-03) SGN=1.0
      IF (ZET.LT.ZETA) GO TO 40
      IF (ZET.GE.ZETA+0.999E-03) GO TO 40
      CALL SELECT (STEP,STPNEW,AINC*PMG)
      KRE=3
50    K2=K2+3
60    CONTINUE
      RETURN
C
      SUBROUTINE SELECT (STEP,STPNEW,STPTRY)
      IF (STEP.GE.0.) STPNEW=AMIN1(STPNEW,STPTRY)
      IF (STEP.LT.0.) STPNEW=AMAX1(STPNEW,STPTRY)
      RETURN
C
      END.

```

REFERENCES

1. I. Flügge-Lotz, "Automatic Control in the Last Two Decades", AIAA Journal, Vol. 10, No. 6, June 1972, pp. 721-726.
2. Tommy W. Case, Jerrel R. Mitchell, W. L. McDaniel, Jr., "A Comparison of Three Computerized Design Algorithms for Control Systems", Proceedings of the 1976 Southeastern Symposium on System Theory, University of Tennessee, Knoxville, Tennessee, 1976.
3. James B. Nail, J. R. Mitchell, W. L. McDaniel, Jr., "Eigenvalue Encouragement Technique", Proceedings of the 1976 Southeastern Symposium on System Theory, Knoxville, Tennessee, April 1976.
4. Anthony Joseph Mancini, "Computer Aided Control System Design Using Frequency Domain Specifications", A Thesis, Naval Postgraduate School, Monterey, California, June 1976.
5. Larry Paul Vines, "Computer Automated Design of Systems", A Thesis, Naval Postgraduate School, Monterey, California, June 1976.
6. Coffey, T. C., "Automatic Frequency Domain Synthesis of Multi-loop Control Systems", AIAA Journal, Volume 8, Number 10, October 1970.
7. Jerrel R. Mitchell, Willie L. McDaniel, Jr., "An Innovative Approach to Compensator Design", NASA Contractor Report, National Aeronautics and Space Administration, Washington, D. C., NASA CR-2248, May 1973.
8. Jerrel R. Mitchell, Willie L. McDaniel, Jr., "Preliminary Studies in the Development of Pole Placement Algorithms", Quarterly Report to the National Aeronautics and Space Administration, Marshall Space Flight Center, Alabama, NAS8-31568, December 1975.
9. Jerrel R. Mitchell, Willie L. McDaniel, Jr., "A Computerized Compensator Design Algorithm with Launch Vehicle Applications", Proceedings of 1976 Southeastern Symposium on System Theory, University of Tennessee, Knoxville, Tennessee, April 1976.
10. Jerrel R. Mitchell, "The Compensator Improvement Program Version VI", Final Report, NASA-ASEE Summer Faculty Fellowship Program, University of Alabama in Huntsville, Huntsville, AL, August 1976.

11. Jerrel R. Mitchell, Willie L. McDaniel, Jr., "Utilizing CIP to Design Compensation For Initially Closed-Loop Unstable Systems", Conference Record of the Ninth Annual Asilomar Conference on Circuits, Systems, and Computers, Monterey, California, Nov 1975.

